VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Informační systém pro správu tras a logů z GPS zařízení

IS for Administration of GPS Tracks and Logs

2018 Bc. Jaroslav Kaštura

Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Jaroslav Kaštura**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T059 Mobilní technologie

Téma: Informační systém pro správu tras a logů z GPS zařízení
IS for Administration of GPS Tracks and Logs

Jazyk vypracování: čeština

Zásady pro vypracování:

Zařízení GPS jsou v dnešní době využívané pro plánovaní tras i pro logování tras projetých. Existuje mnoho výrobců zařízení GPS a také softwarových systémů pro plánovaní tras a správu logů projetých tras (dále jen logů). Tyto zařízení a systémy často disponují pouze omezenou funkcionalitou a navíc používají vlastní rozhraní a formáty dat, přičemž vzájemná konverze není jednoduchá. Cílem této práce je vytvořit informační systém, který bude umožňovat zobrazit a spravovat trasy i logy ve všech používaných formátech.

Úkoly:
1. Nastudovat existující systémy pro zobrazování a správu tras a logů.
2. Nastudovat existující formáty tras a logů GPS zařízení.
3. Nastudovat nejčastější formáty mapových podkladů a jejich vizualizaci.
4. Navrhnout, naimplementovat a otestovat IS pro správu tras a logů, který bude umožňovat:
   a) Zobrazení tras a logů na mapovém podkladu.
   b) Zobrazení tras a logů v textovém formátu.
   c) Editaci tras a logů v textovém editoru.
   d) Konverzi mezi formáty dat tras a logů.
   e) Speciální funkce, např. spojování, rozdělovaní a ořezávaní tras a logů, atd.
   f) Export a import tras v požadovaném formátu.
5. Porovnat vlastní implementaci s existujícími systémy.

Seznam doporučené odborné literatury:

1. Stephen W. Hinch. Outdoor Navigation with GPS. Wilderness Press, Third Edition, 2010.
2. Gabriel Svennerberg. Beginning Google Maps API 3 (Expert's Voice in Web Development). Apress, Second Edition, 2010.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Peter Chovanec, Ph.D.**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2018

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 25. Dubna 2018 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 25. dubna 2018

.....

Děkuji tímto vedoucímu diplomové práce Ing. Petrovi Chovancovi, Ph.D. za odbornou pomoc, ochotu a konzultace při realizaci tohoto projektu. Dále bych rád poděkoval Ing. Martině Litschmannové, Ph.D. a Ing. Janu Kracíkovi, Ph.D. za velmi užitečné informace z oblasti statistiky a pravděpodobnosti. Ing. Radimu Bačovi, Ph.D. za informace o administraci databázových systémů, Ing. Mgr. Michalu Krumniklovi, Ph.D. a Ing. Pavlu Moravcovi, Ph.D. za informace z oblasti mobilních technologií.

Abstrakt

Motivem pro vytvoření této práce byla absence intuitivního nástroje pro správu a vizualizaci tras a logů z GPS zařízení. Pro řešení tohoto problému bylo potřeba vytvořit informační systém umožňující přehlednou vizualizaci logů z GPS zařízení, správu tras a konverzi mezi formáty. Vyvinutý systém pracuje s běžně používanými formáty logů z GPS zařízení, které převede do interního datového formátu umožňujícího vizualizualizaci na mapě nebo export do jiných formátů. Správa tras je řešena pomocí uživatelského rozhraní, kde uživatel informačního systému zvolí způsob prezentace trasy v mapě a může upravovat metadata trasy. Dále informační systém umožňuje funkce jako korekci trasy v textovém editoru, spojování různých tras, rozdělení trasy nebo její ořezání.

Klíčová slova: Informační systém, vizualizace GPS logů, elektronické mapové podklady, GPS zařízení, lokalizace objektů

Abstract

The motivation for this project was the absence of intuitive tool for visualization and management of paths and logs from GPS devices. To solve these problems, there was a need to develop a software that will be able to create clear visualisations based on GPS device's logs, route management and data format conversions. The software works with common data types from GPS devices, which are converted to a proprietary data format allowing visualization on a map or export to other formats. Route management contains an user interface, where the user can select a way how to visualise the data on a map and change the route metadata. The software also allows other functions, such as route corrections in a text editor, route merging, splitting or trimming.

Key Words: Information system, ASP.NET technology, eletronic map data, GPS device, object localization

Obsah

Seznam použitých zkratek a symbolů

GPS	–	Globální Polohový Systém
IS	–	Informační Systém
HTML	–	Hyper Text Markup Language
HTTP	–	HyperText Transfer Protocol
ORM	–	Objektově Relační Mapování
SŘDB	–	Systém Řízení Báze Dat
JS	–	JavaScript
SPA	–	Single Page Application
OGC	–	Open Geospatial Consorcium
GIS	–	Geografický Informační Systém
CLI	–	Command Line Interface
MDB	–	Material Design Bootstrap
AJAX	–	Asynchronní Javascript A Xml
DevTools	–	The Chrome Developer Tools

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Lokalizace, neboli určování umístění objektu v geoprostoru je běžně poskytovaná funkcionalita moderních mobilních zařízení. Existuje celá řada způsobů, pomocí nichž mobilní zařízení získávají informace o aktuální pozici objektu. Tím nejznámějším je lokalizace pomocí GPS (z anglického Global Positioning System) [29] modulu, který bývá často doplňován pomocnými systémy, jako je například lokalizace pomocí rádiových sítí Wifi (z anglického Wireless Fidelity) [35] nebo GSM (z francouzského Groupe Spécial Mobile) [50].

Využívat lokalizační údaje lze mnoha způsoby, ať už k navigaci nebo lokalizaci různých objektů. Přesnost určení polohy lze v součinnosti s dalšími metodami zvýšit až na jednotky centimetrů, kdy takovou službu mohou volně využívat i běžní uživatelé bez jakéhokoli omezení.

K čemu je to všechno potřeba? V dnešní době jsme neustále v pohybu – autem, pěšky, na kole, běháme atd. Navštívíme zajímavá místa, překonáme různé terény, zdoláme výšková převýšení, objevíme nové cesty. V případě, že tento pohyb zaznamenáme pomocí GPS zařízení, můžeme ze získaných dat za pomocí různých informačních systémů provést analýzu trasy nebo jí porovnat s jinými trasami. Rovněž můžeme trasu upravovat a sdílet s přáteli.

Tuto diplomovou práci jsem si vybral, protože bych rád doplnil stávající informační systémy ke správě tras a logů GPS zařízení o informační systém, který bude k této problematice přistupovat odlišným způsobem.

2 Základy lokalizace

Matematický popis povrchu země není triviální jako matematický popis elipsoidu nebo koule, protože povrch země je velmi nepravidelný, členitý a složitý. Z tohoto důvodu je nutné reálný povrch planety země aproximovat zjednodušeným modelem.

2.1 Geoid

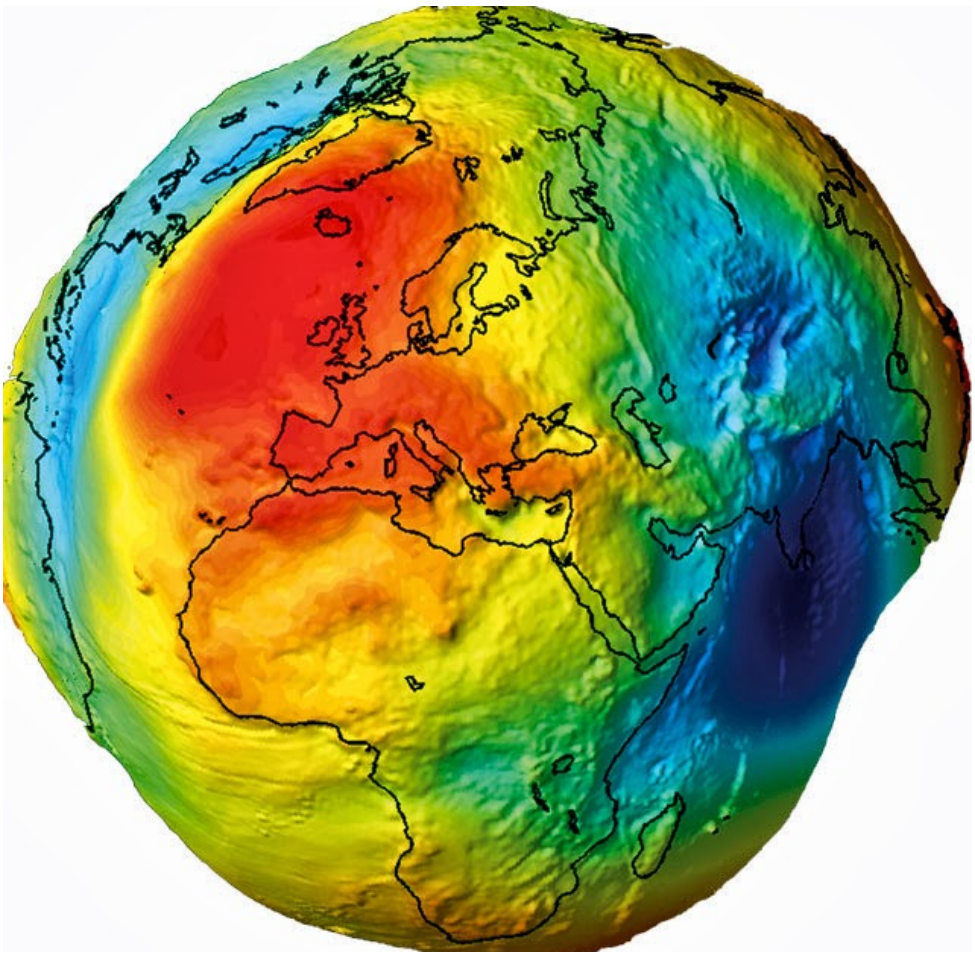
Definice 1 *Geoid [8] je myšlená nulová hladinová ekvipotenciální plocha, která je v každém bodě kolmá na směr zemské tíže.*

Stejně jako povrch země je i geoid těleso nepravidelné, členité a velmi složité, proto jej rovněž nelze jednoduše aproximovat a pro potřeby mapování jej nahrazujeme referenčními plochami. Přibližný tvar geoidu je zobrazen na obrázku č.1.

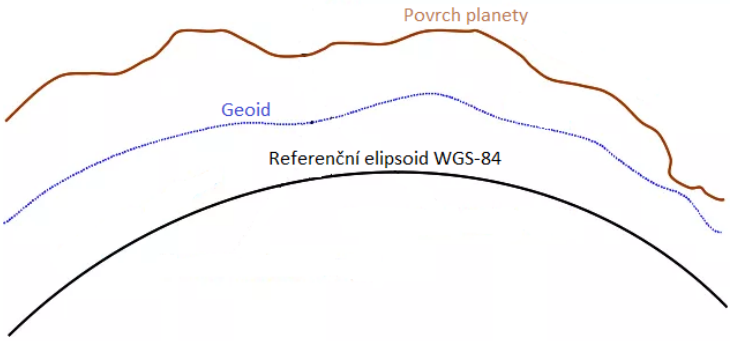
2.2 Referenční plochy

Mezi referenční plochy patří referenční rovina, koule a elipsoid. Za referenční rovinu [8] lze považovat zemský povrch kruhového tvaru o poloměru do 8 km. Pokud se jedná o území větší, jsou výpočty nepřesné. Proto se rovinou nebudu zabývat. Referenční koule [8] se používá pro geodetické a kartografické účely a to tak, že se zeměpisné souřadnice užijí na kouli. Takovým zobrazením ovšem dochází k velkým deformacím délek, proto se referenční kouli nebudu v této práci zabývat.

Referenční elipsoid [8] je nejpřesnější způsob mapování zemského povrchu. Mezi referenční elipsoidy patří WGS-84 [8], NAD83 [8] nebo Krakovského elipsoid [8], přičemž se dnes nejčastěji používá referenční elipsoid WGS-84. Na obrázku č. 2 lze vidět srovnání skutečného tvaru země, referenčního elipsoidu WGS-84 a geoidu. Na konverze geografických souřadnic mezi různými typy referenčních ploch jsou k dispozici hotová volně dostupná řešení jako je například projekt PROJ.4 [24]. Ten byl původně vyvíjen jako filtr pro Unixové operační systémy na konverzi geografických souřadnic zeměpisné šířky a délky do kartézské souřadnicové soustavy. Dnes se tato knihovna využívá ke konverzi geografických souřadnic mezi všemi typy referenčních ploch.



Obrázek 1: Vizualizace geoid



Obrázek 2: Srovnání skutečného povrchu planety, referenčního elipsoidu WGS-84 a geoid

3 Formáty logů

Zařízení zaznamenávající trasu objektu se vyvíjejí již desítky let a s jejich novými možnostmi rostou požadavky na datové formáty umožňující uchovávat detailnější informace z monitorované trasy. V této kapitole budou popsány datové struktury a schémata, která se v dnešní době používají pro efektivní ukládání takovýchto komplexních dat. Všechny níže popsané formáty používají referenční elipsoid WGS-84.

Definice 2 *Track log je soubor dodaný navigačním systémem, obsahující časové a prostorové informace o místech výskytu sledovaného subjektu.*

3.1 GPX

GPX [25] nebo také GPS Exchange formát, je XML schéma navržené pro ukládání lokalizačních dat. Lze jej využívat k popisu trasy, zastávek, nadmořské výšky a času v bodech trasy. Formát je snadno rozšiřitelný o nové atributy a elementy, jako jsou například rychlost nebo směr. V ukázce č. 1 lze vidět schéma GPX souboru. V atributech gpx elementu jsou reference na schémata tohoto xml dokumentu a také reference na rozšíření, které jsou v souboru použity. Tyto informace jsou využívány při vyhledávání požadovaných elementů v dokumentu. Dále je zde prostor k uložení sumárních dat o trase.

```
<?xml version='1.0' encoding='UTF-8' standalone='yes' ?>
<gpx version="1.1" xmlns="http://www.topografix.com/GPX/1/1" xmlns:geotracker="http://
    ilyabogdanovich.com/gpx/extensions/geotracker" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    instance" xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com/
    GPX/1/1/gpx.xsd" creator="Kastan">
  <metadata>
    <name>Ukazka</name>
    <author>
      <name>Created by Gipsy</name>
      <link href="https://gipsy.cz" />
    </author>
    <link href="https://gipsy.cz" />
    <time>2018-03-04T13:24:44.921Z</time>
  </metadata>
  <trk>
    <name>Ukazka</name>
    <src>Created by Gipsy</src>
    <link href="https://gipsy.cz" />
    <extensions>
      <geotracker:meta>
        <length>2206.15220440182</length>
        <duration>16848910000</duration>
        <creationtime>2018-03-01T12:19:56.610Z</creationtime>
        <activity>0</activity>
```

```
      </geotracker:meta>
    </extensions>
    <trkseg>
      <trkpt lat="50.034134" lon="17.307695">
        <ele>731</ele>
        <time>2017-03-21T16:41:57.000Z</time>
      </trkpt>
    </trkseg>
  </trk>
</gpx>
```

Výpis 1: Ukázka GPX formátu

3.2 Garmin course CRS a TCX

Garmin course CRS [25] byl první verzí schématu pro ukládání logů a dnes se používá pouze u starých zařízení Garmin. Training Center XML nebo také TCX [25] je formát společnosti Garmin uvedený v roce 2007 jako součást produktů této společnosti. Jak je patrno z ukázky č. 2, velice se podobá GPX formátu, avšak s tím rozdílem, že jsou ve standardu navíc definovány elementy pro srdeční tep, kadence při běhu nebo jízdě na kole nebo informace o spálených kaloriích. Stejně jako u předchozího formátu, i tady je prostor k uložení sumárních dat o trase. Za výhodu lze považovat skutečnost, že nejsou potřeba žádná rozšíření, protože všechny důležité elementy a atributy jsou definovány již v základním schématu.

```
<?xml version="1.0"?>
<TrainingCenterDatabase xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
    www.w3.org/2001/XMLSchema" xmlns="http://www.garmin.com/xmlschemas/TrainingCenterDatabase">
  <Workouts />
  <Courses />
  <Activities>
    <Activity Sport="Biking">
      <Lap StartTime="0001-01-01T00:00:00">
      <Track>
        <Trackpoint>
          <Time>2017-03-21T16:41:57</Time>
          <Position>
            <LatitudeDegrees>50.034134</LatitudeDegrees>
            <LongitudeDegrees>17.307695</LongitudeDegrees>
          </Position>
          <AltitudeMeters>731</AltitudeMeters>
          <HeartRateBpm>
            <Value>0</Value>
          </HeartRateBpm>
          <Extensions />
        </Trackpoint>
      </Track>
      <TotalTimeSeconds>0</TotalTimeSeconds>
```

```
      <DistanceMeters>0</DistanceMeters>
      <Calories>0</Calories>
      <Intensity>Active</Intensity>
      <TriggerMethod>Manual</TriggerMethod>
    </Lap>
    <Id>2018-03-04T14:24:45.9306979+01:00</Id>
    <Creator xsi:type="Application_t">
      <Name>Kastan</Name>
    </Creator>
    </Activity>
  </Activities>
</TrainingCenterDatabase>
```

Výpis 2: Ukázka TCX formátu

3.3 FIT (ANT+)

Je flexibilní datový protokol navržen k ukládání a sdílení dat ze sportovních, zdravotních a fitness aktivit. FIT protokol [25] definuje sadu šablon pro ukládání dat, které lze použít k uchování takových informací, jako jsou uživatelské profily a údaje o aktivitách jejich uživatelů. Navrhnut je tak, aby byl kompaktní, interoperabilní a rozšiřitelný. Možnosti tohoto formátu jsou velmi komplexní, proto zde není prezentován ukázkový kód.

3.4 KML

Keyhole Markup Language [25] je otevřený standard též pojmenovaný OpenGIS Encoding Standard (OGC KML)[60]. Nejen proto, že se tento formát stal v roce 2008 standardem OGC, ale i z důvodu velmi komplexního schématu umožňujícího práci s téměř všemi druhy prostorových dat, je tento formát velmi často využíván tvůrci GIS softwaru. Byť je toto schéma velmi obsáhlé, pro práci s track logy je mnohdy využito rozšíření, v němž jsou uloženy údaje o rychlosti a směru objektu, jak lze vidět v ukázce č. 3. Častokrát se jedná o redundantní informace, které lze poměrně snadno dopočítat.

```
<?xml version='1.0' encoding='UTF-8' standalone='yes' ?>
<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:gx="http://www.google.com/kml/ext/2.2" xmlns:
    atom="http://www.w3.org/2005/Atom">
<Document>
  <open>1</open>
  <visibility>1</visibility>
  <name>trasa.kmz</name>
  <atom:author>
    <atom:name>Created in Gipsy by Kastan</atom:name>
  </atom:author>
  <atom:link href="https://gipsy.cz" />
  <TimeStamp>
    <when>2018-03-01T12:19:56.610Z</when>
```

```
  </TimeStamp>
  <Style id="track">
    <LineStyle>
      <color>ffff0000</color>
      <width>4</width>
    </LineStyle>
  </Style>
  <Schema id="geotrackerTrackSchema">
    <gx:SimpleArrayField name="speed" type="float">
      <displayName>Speed, meters per second</displayName>
    </gx:SimpleArrayField>
  </Schema>
  <Placemark id="tour">
  <name>trasa.kmz</name>
  <description></description>
  <atom:author>
    <atom:name>Created in Gipsy by Kastan</atom:name>
  </atom:author>
  <atom:link href="https://gipsy.cz" />
  <styleUrl>#track</styleUrl>
  <gx:MultiTrack>
    <altitudeMode>absolute</altitudeMode>
    <gx:interpolate>0</gx:interpolate>
      <gx:Track>
        <gx:coord>17.307695 50.034134 731</gx:coord>
        <when>2017-03-21T16:41:57.000Z</when>
        <gx:coord>17.307693 50.034267 708</gx:coord>
        <when>2017-03-21T16:42:00.000Z</when>
        <ExtendedData>
          <SchemaData schemaUrl="#geotrackerTrackSchema">
            <gx:SimpleArrayData name="speed">
              <gx:value></gx:value>
            <gx:value>4.96357515149577</gx:value>
            </gx:SimpleArrayData>
          </SchemaData>
       </ExtendedData>
    </gx:Track>
  </gx:MultiTrack>
</Placemark>
</Document>
</kml>
```

Výpis 3: Ukázka KML formátu

3.5 KMZ

Keyhole Markup Language Zipped [25] je komprimovanou verzí KML souboru, který lze doplnit o další soubory. Těmi mohou být například textury objektů definovaných v KML souboru.

3.6 PCX5

Garmin PCX5 [25] je velmi jednoduché schéma, kde jsou nejprve uvedeny metadata o trase a následně informace o jednotlivých bodech trasy, čas a nadmořská výška, jak lze vidět ve výpisu kódu č.4. Výhodou tohoto formátu je absence redundancí, díky čemu se zmenšuje objem dat výsledného souboru. Nevýhodou je nemožnost rozšíření schématu.

```
H  SOFTWARE NAME & VERSION
I  PCX5 2.08 Generated by GPSies.com

H  R DATUM              IDX DA          DF          DX          DY          DZ
M  G WGS 84             121 +0.000000e+00 +0.000000e+00 +0.000000e+00 +0.000000e+00 +0.000000e
    +00

H  COORDINATE SYSTEM
U  LAT LON DEG

H  TN Kmlimporttest2018

H  LATITUDE   LONGITUDE   DATE     TIME     ALT
T  +50.0830040 +14.4369080 01-JAN-10 00:00:00 00218
T  +50.0830920 +14.4369710 01-JAN-10 00:00:03 00233
T  +50.0831600 +14.4368830 01-JAN-10 00:00:07 00231
```

Výpis 4: Ukázka PCX5 formátu

3.7 CSV

CSV [25] nebo také Comma-Separated Values je nejminimalističtější schéma na ukládání track logů. Jak lze vidět ve výpisu kódu č. 5, jde pouze o hodnotu zeměpisné šířky, zeměpisné délky a nadmořské výšky, které jsou odděleny čárkou.

```
Latitude,Longitude,Elevation
50.08300400,14.43690800,218.0
50.08309200,14.43697100,233.0
50.08316000,14.43688300,231.0
```

Výpis 5: Ukázka CSV formátu

3.8 TomTom ITN formátu

TomTom ITN [25] je další minimalistické schéma, které uchovává pouze hodnotu zeměpisné šířky a zeměpisné délky. Přes pokus o maximální minimalizaci datového objemu, schéma obsahuje na

každém řádku redundantní informaci atributu POINT. Ve výpise kódu č. 6 je možné vidět ukázku ITN formátu.

```
Latitude,Longitude,Elevation
1443690|5008300|POINT 1-1|0|
1443697|5008309|POINT 1-2|0|
1443688|5008316|POINT 1-3|0|
```

Výpis 6: Ukázka ITN formátu

3.9 JSON Track

V JSON [20] je trasa uložena v objektu typu MultiLineString a skládá se z jednotlivých bodů uložených v poli souřadnic. Ukázku formátu je možné vidět ve výpisu kódu č. 7. JSON objekt je doplněn o metadata, jako je název trasy nebo její popis.

```
{
   "type": "FeatureCollection",
   "features": [
      {
         "type": "Feature",
         "geometry": {
            "type": "MultiLineString",
            "coordinates": [[[14.436908, 50.083004, 218.0],
                             [14.436971, 50.083092, 233.0],
                             [14.436883, 50.08316, 231.0]]],
            "bbox": [16.97233, 50.102268, 14.436705, 49.757034]
         },
         "properties": {
            "name": "Trasa",
            "desc": "Generated by gipsy.com http://www.gipsy.cz/"
         }
      }
   ]
}
```

Výpis 7: Ukázka JSON formátu

3.10 Shrnutí

V této kapitole byly popsány nejpoužívanější formáty track logů, včetně jejich výhod a nevýhod. V první fázi vývoje bude informační systém podporovat formáty KML, KMZ, GPX a TCX, protože jsou v dnešní době používány nejčastěji. Pro zjednodušení práce s track logy bude výhodné implementovat vlastní univerzální datovou strukturu, která bude kompatibilní se všemi formáty. Bude-li v budoucnu zájem rozšířit informační systém o formáty, které používají jiný referenční elipsoid než WGS-84, lze ke konverzi mezi souřadnicovými systémy použít knihovnu PROJ.4 [24].

4 Mapové nástroje a podklady

V dnešní době je k dispozici velké množství nástrojů umožňujících práci s mapovými podklady a není snadné vybrat z nich ten nejvhodnější. V této kapitole budou rozebrány nejznámější mapové podklady a také výhody a nevýhody nástrojů dovolujících práci s nimi.

4.1 OpenLayers

OpenLayers [61] umožňuje velmi snadno integrovat dynamické mapy do jakékoli webové stránky. Lze zobrazovat mapové dlaždice, vektorová data a značky z jakéhokoli formátu. OpenLayers byla vyvinuta k prezentaci většiny geografických informací. Knihovna je zcela zdarma pod licencí FreeBSD [70]. Práce s tímto nástrojem je velmi snadná. V HTML dokumentu deklarujeme div element s jedinečným id, které vložíme jako argument do konstruktoru Openlayer, a ten už zajistí vykreslení mapových podkladů v elementu. Ukázku kódu můžete vidět ve výpisu kódu č.8. Dále stojí za povšimnutí vlastnost `interactions` v konstruktoru objektu `ol.Map`, díky které můžeme velmi snadno nástroj rozšířit o další funkcionalitu. V tomto případě se jedná o rozšíření ovládacího panelu o funkcionalitu `DragRotateAndZoom` [61].

```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <link rel="stylesheet" href="https://openlayers.org/en/v4.6.4/css/ol.css" type="text/css">
    <script src="https://openlayers.org/en/v4.6.4/build/ol.js"></script>
  </head>
  <body>
    <div id="map" class="map"></div>
    <script>
      var map = new ol.Map({
      interactions: ol.interaction.defaults().extend([
          new ol.interaction.DragRotateAndZoom()
        ]),
        layers: [
          new ol.layer.Tile({
            source: new ol.source.OSM()
          })
        ],
        target: 'map',
        view: new ol.View({
          center: [0, 0],
          zoom: 2
        })
      });
    </script>
  </body>
</html>
```

Výpis 8: Ukázka zobrazení mapy pomocí nástroje Openlayers

4.2 Leaflet

Stejně jako knihovna Openlayers, pracuje knihovna Leaflet [62] s dynamickými mapovými podklady. Jedná se o opensource knihovnu, jejíž zdrojové kódy jsou velmi pěkně zdokumentovány a jsou dostatečně přehledné. Leaflet se pyšní mobile-friendly funkcionalitou a proto je vhodnější pro responsivní aplikace. Je navržena pro efektivní práci na většině běžně používaných desktopových i mobilních platformách a lze jí rozšířit o velké množství komponent. Nasazení komponenty je velmi podobné nástroji OpenLayers, jak je patrno z výpisu kódu č. 9. Opět je na stránce div s jedinečným id, které je předáno nástroji Leaflet jako argument. Navíc lze v kódu vidět, jakým způsobem mohou být přidávány nové objekty do mapy (`L.marker([51.5, -0.09]).addTo(mymap)`), případně jak vytvořit posluchač událostí (`mymap.on('click', onMapClick)`), který v tomto případě spustí funkci `onMapClick(e)` a v parametru `e` předá podrobné informace o události.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <link rel="stylesheet" href="https://unpkg.com/leaflet@1.3.1/dist/leaflet.css"/>
   <script src="https://unpkg.com/leaflet@1.3.1/dist/leaflet.js"></script>
  </head>
  <body>
    <div id="mapid"></div>
    <script>
      var mymap = L.map('mapid').setView([51.505, -0.09], 13);
      var marker = L.marker([51.5, -0.09]).addTo(mymap);
      function onMapClick(e) {
         alert("You clicked the map at " + e.latlng);
      }
      mymap.on('click', onMapClick);
    </script>
  </body>
</html>
```

Výpis 9: Ukázka zobrazení mapy pomocí nástroje Leaflet

4.3 Google Maps

Google Maps [65] ke svým mapovým podkladům poskytuje knihovny pro všechny běžně používané frameworky prezentační vrstvy. Všechny knihovny jsou přehledné a snadno použitelné. Mapové podklady jsou velmi detailní a poskytují informace o dopravě. Dopravní informace v

Google mapách koresponduje se skutečnou dopravní situací a to především díky mobilním zařízením, které odesílají anonymní informace o dopravní situaci a společnost Google tyto informace vyhodnocuje pomocí umělé inteligence. Nástroje Google Maps poskytují mnoho funkcí ke stylování map, navigaci nebo informaci o vybrané lokalitě. Práce s těmito mapami je stejně snadná jako v předchozích případech, jak můžete vidět ve výpisu kódu č. 10. Další výhodou je možnost využívat Google Elevation API [80], díky kterému je snadné provádět korekci výšky v track logu.

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <h3>My Google Maps Demo</h3>
    <div id="map"></div>
    <script>
      function initMap() {
        var uluru = {lat: -25.363, lng: 131.044};
        var map = new google.maps.Map(document.getElementById('map'), {
          zoom: 4,
          center: uluru
        });
        var marker = new google.maps.Marker({
          position: uluru,
          map: map
        });
      }
    </script>
    <script async defer
    src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap">
    </script>
  </body>
</html>
```

Výpis 10: Ukázka zobrazení mapy pomocí nástroje Google Maps

4.4 Openstreet Maps

Openstreet Maps [63] spravuje komunita nadšenců a profesionálů. Tito lidé mohou editovat nebo přidávat data o silnicích, železničních trasách, zeleních a všech objektech podporovaných v tomto softwaru. Tyto mapy jsou v některých místech detailněji propracovány než mapy Googlu. Bohužel obsahují i řadu nepřesností, které mohou způsobit problémy v případě profesionálního použití. Protože do těchto map může přispívat téměř kdokoli, nejsou vhodné pro aplikace vyžadující přesnost a pravdivost dat. Komunita, která tvoří tyto mapové podklady, využívá pro vizualizaci dat na vlastním webu knihovnu Leaflet.

4.5 Bing Maps

Bing Maps [64], jsou vytvořené a spravované společností Microsoft [81] a poskytují jak mapové podklady, tak knihovny pro práci s nimi. Tyto mapy lze velmi snadno integrovat do všech technologií společnosti Microsoft a nejen zde. Jsou dostupné dostatečně kvalitní a přehledné knihovny a wrapery pro nejpoužívanější javascriptové a typescriptové frameworky. V mapových podkladech jsou velmi dobře zmapovány silnice a poskytují také informace o dopravní situaci. Bohužel dopravní situace mnohdy nekoresponduje se skutečnou dopravní situací. Za největší nevýhodu lze považovat málo detailní mapové podklady (například chybějící vizualizace budov). Ceny za využívání Bing Maps a knihoven se liší v závislosti na způsobu využití.

4.6 Shrnutí

Všechny výše uvedené knihovny pro práci s mapovými podklady jsou velmi propracované a vhodné pro vizualizaci prostorových dat. Mapové podklady Openstreet Maps a Google jsou velmi detailní a přehledné, což o mapách Bing Maps nelze tvrdit. V případě Google Maps jsou k dispozici velmi propracované knihovny pro práci s těmito mapovými podklady a to pro všechny běžně používané platformy. V budoucnu plánuji funkcionalitu vyvíjené aplikace rozšiřovat o funkce, které jsou integrovány pouze v nástrojích Google Maps. Jedná se o nástroje Google Elevation API nebo navigaci, která bude reflektovat dopravní situaci. Z těchto důvodů jsem se rozhodl pro použití mapových podkladů Google Maps.

5 Systémy pro zobrazení a správu tras a logů

Problematika zpracování a vizualizace tras a logů je velmi složitá a na trhu funguje už celá řada informačních systémů, které se snaží pokrýt všechny požadavky koncových uživatelů. V této kapitole bude proveden průzkum existujících informačních systémů určených pro správu tras a logů, a budou popsány jejich výhody a nedostatky.

5.1 GPSies

GPSies [30] je velmi sofistikovaná aplikace na zpracování track logů, která se skládá z těchto komponent:

- Plánovač tras slouží k navrhování tras, přičemž uživatel určí průjezdové body a aplikace najde nejkratší cestu mezi těmito body, vypočítá délku a její převýšení. Při editaci lze vybrat dopravní prostředek, který bude uživatel při zdolání trasy využívat. Bohužel volba dopravního prostředku neovlivní výběr trasy mezi dvěma body. Poslední variantou je možnost kopírování silnice při přidání nového bodu, což znamená, že je automaticky nalezena cesta po silnici. Samozřejmostí je import a export trasy, kde lze využít všechny známé GPS formáty.
- Prohlížeč trasy je jednou z nejtriviálnějších částí této aplikace. Trasy jsou zde prezentovány značně přehledně a mapy s trasou jsou doplněny přehlednými grafy rychlostí a převýšení v bodech trasy, jak lze vidět na obrázku č. 3.
- Konvertor formátů viz. obrázek č. 4, slouží ke konverzi souboru z jednoho formátu do jiného. Tento informační systém podporuje všechny formáty popsané v kapitole č. 3 a mnoho dalších.

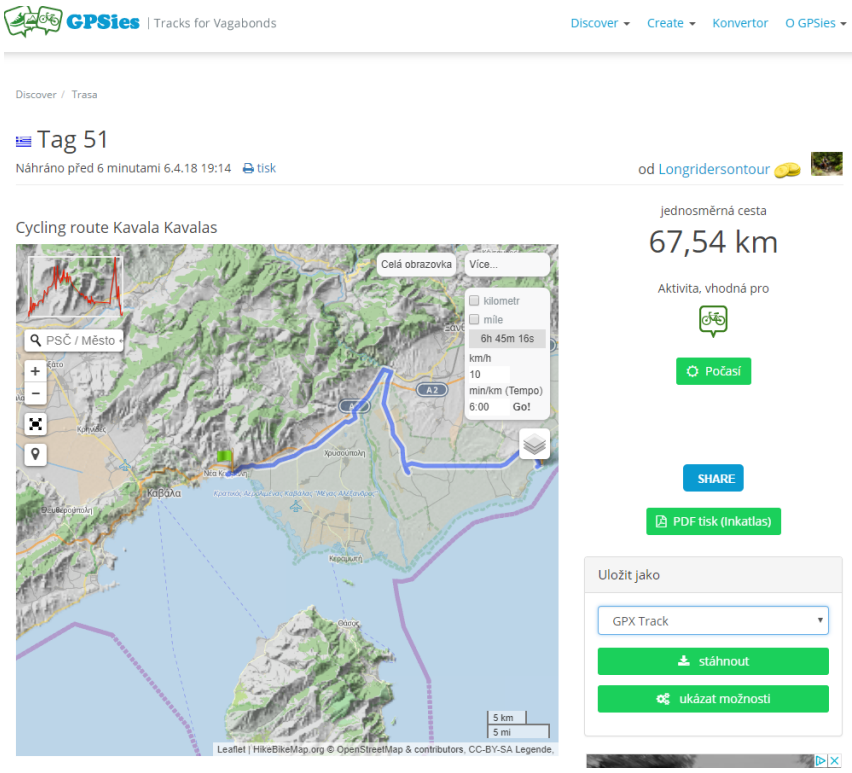
Aplikace využívá knihovnu Leaflet JS pro vizualizaci mapových podkladů a mapové podklady OpenStreet Maps.

5.2 GPS visualizer

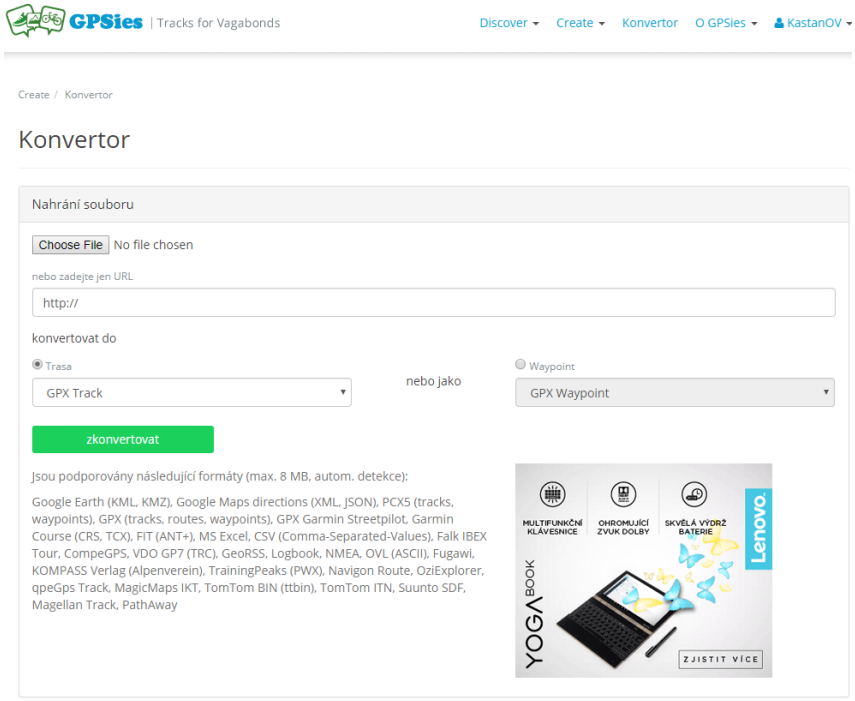
GPS visualizer [32] dokáže vizualizovat všechny běžně používané track logy. Aplikace pracuje s Google Maps a její funkcionalita je omezená na vizualizaci jednotlivých souborů, jak lze vidět na obrázku č. 5. Trasy lze exportovat do rastrového obrázku jpg nebo bmp. Dále můžeme trasy exportovat do formátu GPX, což je jediný podporovaný formát aplikace.

5.3 Maplorer

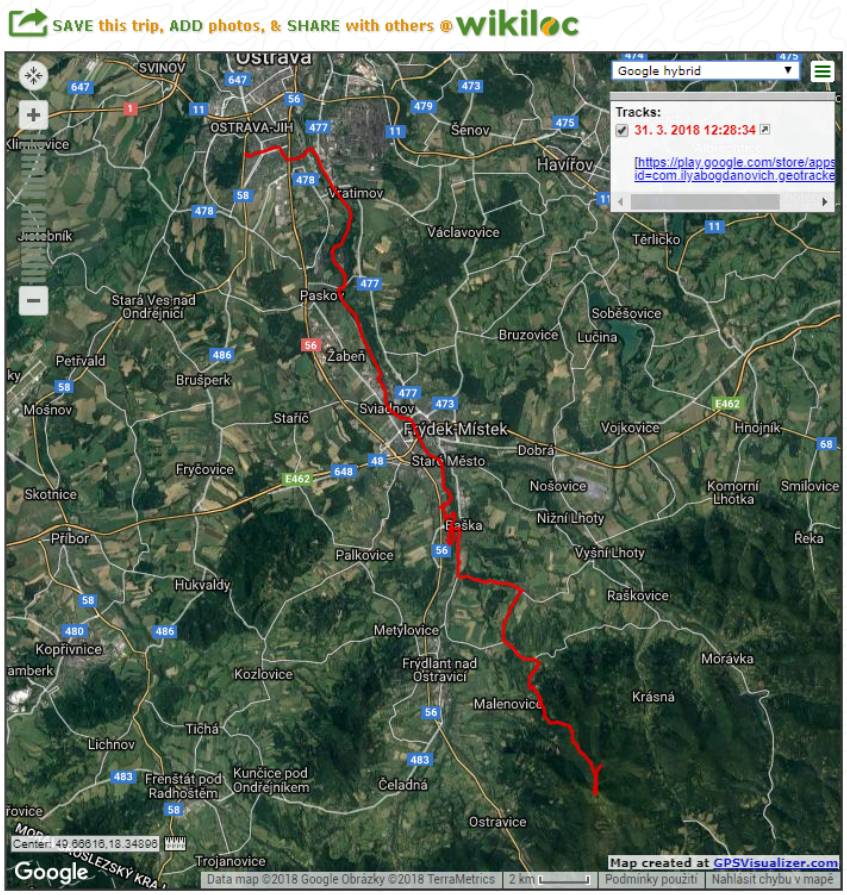
Maplorer [38], který lze vidět na obrázku č. 6, je poměrně jednoduchý software pracující s GPX schématem, jenž využívá Google Maps k vizualizaci trasy. Mapa je doplněna o graf s nadmořskou výškou z GPX souboru a druhý graf s nadmořskou výškou získanou z Google



Obrázek 3: Ukázka aplikace GPSies

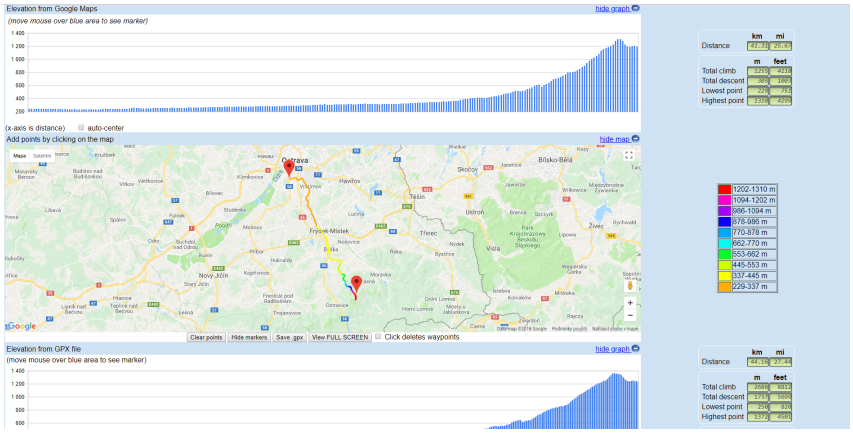


Obrázek 4: Ukázka GPSies konvertoru



Obrázek 5: Ukázka aplikace GPS visualizer

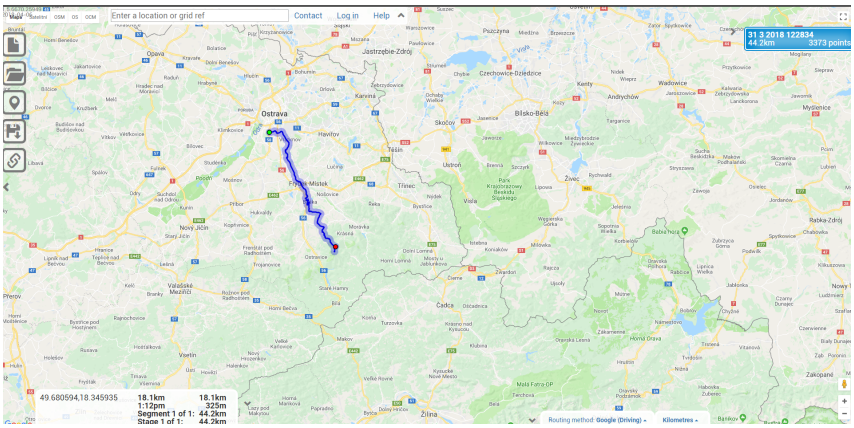
Elevation API. Aplikace obsahuje řadu nefunkčních nebo nedokončených komponent, jako je například GPXview3D.



Obrázek 6: Ukázka aplikace Maplorer

5.4 GPX editor

GPX editor [33], který lze vidět na obrázku č. 7, využívá mapové podklady Google Maps, OpenStreets Maps. Aplikace pracuje pouze se soubory GPX a CSV, které lze vizualizovat na všech výše uvedených mapových podkladech. O vykreslení mapových podkladů se starají knihovny Googlu. Za nevýhodu této aplikace lze považovat, že při nahrání trasy do aplikace si musí uživatel vyhledat trasu v mapě ručně. Naopak výhodou je možnost vytvoření nové trasy nebo editace uložených tras.



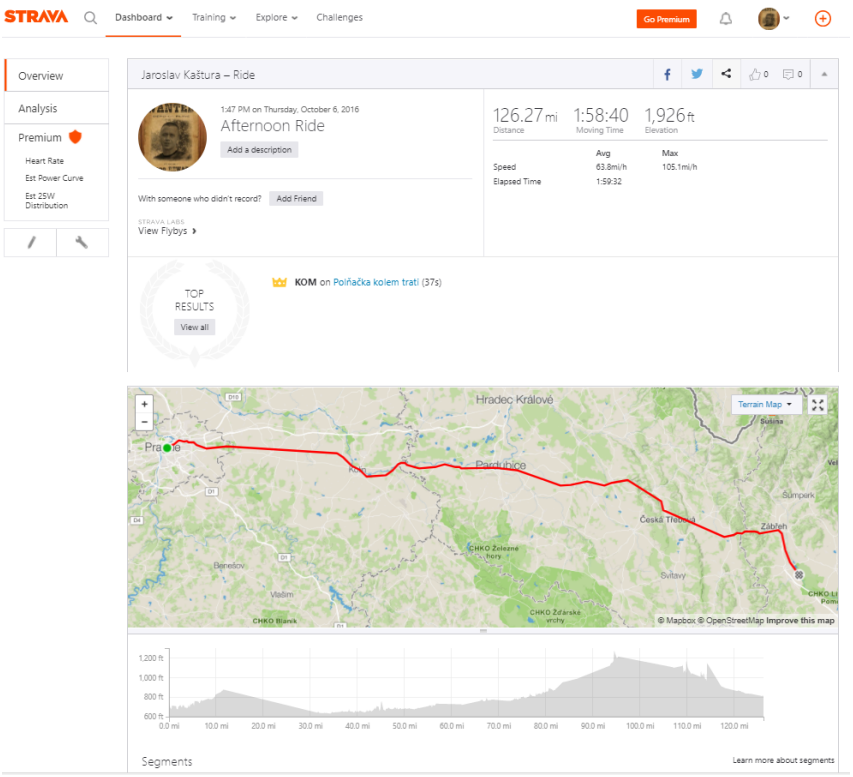
Obrázek 7: Ukázka aplikace GPX editor

5.5 Strava

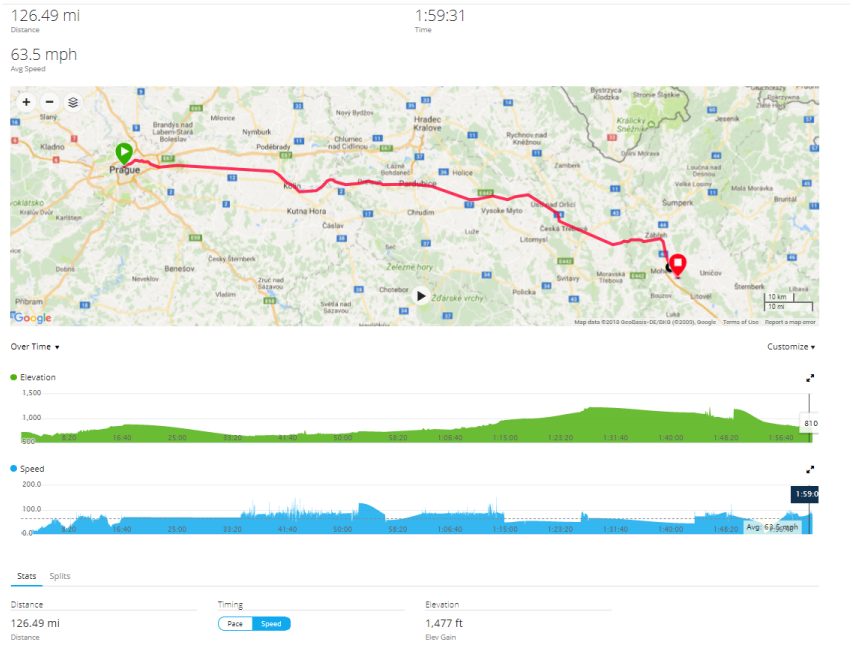
Strava [39] je velmi komplexní nástroj pro fitness a sport, kde lze trasy plánovat, upravovat, spojovat, rozdělovat a vizualizovat. Vizualizace trasy je doplněna o graf převýšení a je možné ji sdílet na sociálních sítích nebo odkazem v emailu. Ukázku aplikace Strava lze vidět na obrázku č. 8. Po načtení trasy je vypočítána její celková vzdálenost a celkový čas. Dále jsou zde zobrazeny údaje o průměrné a maximální rychlosti. Systém obsahuje analýzu trati včetně rychlostního grafu. Pokud logovací zařízení podporuje monitorování srdečního tepu, je tento údaj rovněž graficky prezentován uživateli. Výrobce poskytuje mobilní aplikace pro IOS a Android, které umožňují řídit trénink nebo nahrávat trasu bez využití speciálního zařízení.

5.6 Garmin Connect

Služba Garmin Connect [40] je online tréninkový nástroj pro ukládání, analýzu a sdílení fitness aktivit. Tato služba je určena především pro uživatele využívající zařízení Garmin. Aplikace využívá mapové podklady Google Maps, včetně rozšíření Google Elevation API. Jsou zde velmi přehledně vizualizovány trasy, a to včetně grafů s převýšením a rychlostí. Konverze je podporována pouze mezi formáty společnosti Garmin, což lze považovat za nedostatek této aplikace, ale zároveň skvělý marketingový tah. Ukázku aplikace lze vidět na obrázku č. 9.



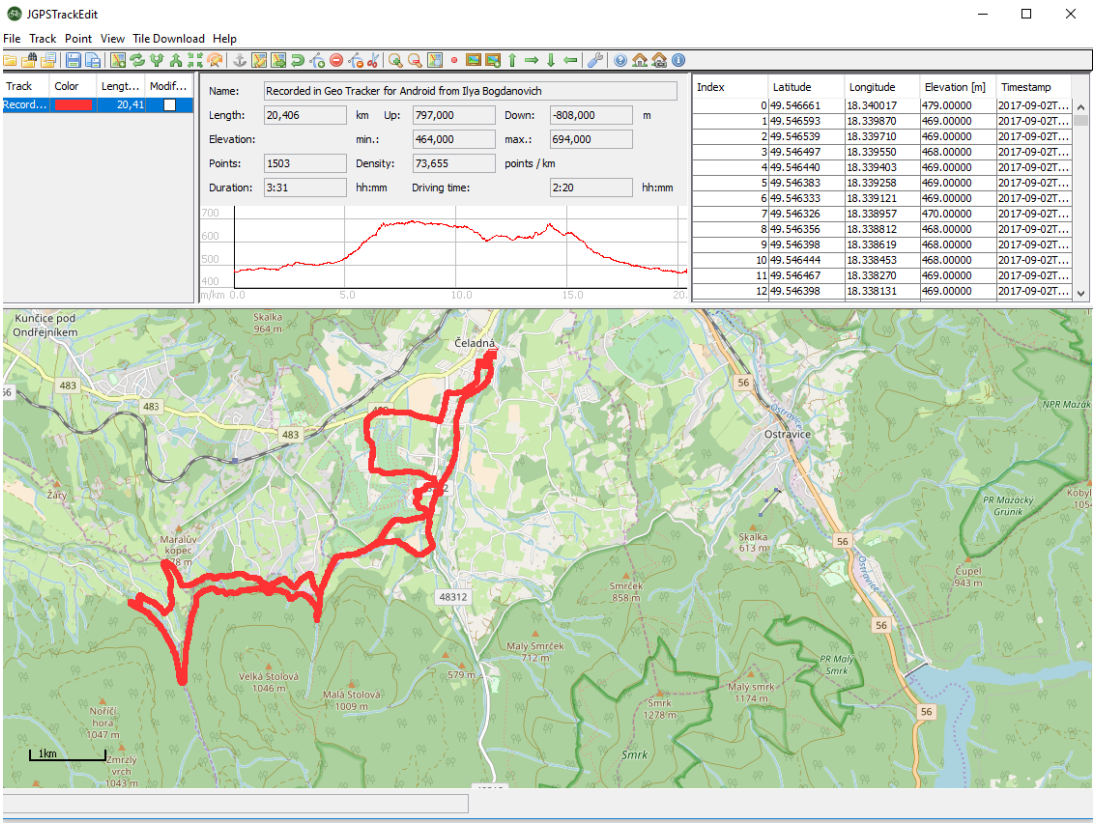
Obrázek 8: Ukázka aplikace Strava



Obrázek 9: Ukázka aplikace Garmin connect

5.7 JGPS track editor

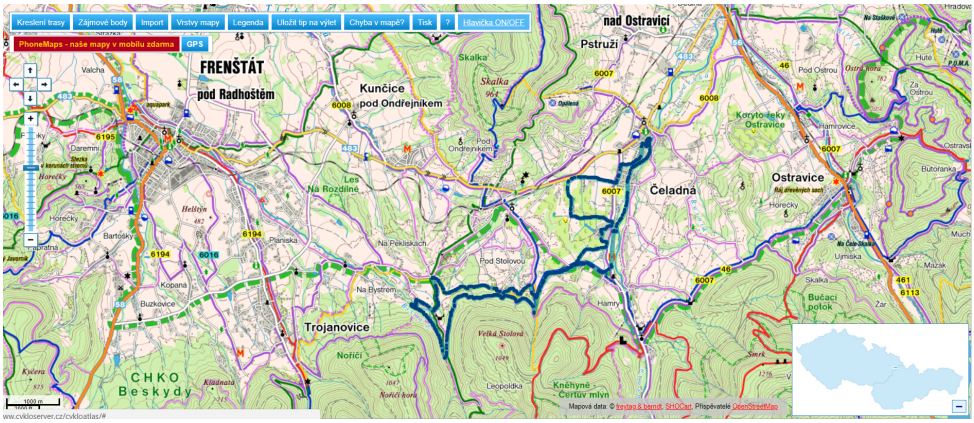
JGPS track editor [37] je multiplatformní aplikace, která poskytuje nástroje pro vizualizaci tras, jejich vytváření a editaci. Ukázku aplikace lze vidět na obrázku č. 10. Dále je podporována funkcionalita spojování, rozdělování tras a konverze mezi formáty FIT, GPX, TCX a KML. Podporovány jsou různé mapové podklady, přičemž ve výchozím nastavení jsou používány mapové podklady OpenStreets Maps.



Obrázek 10: Ukázka aplikace JGPS track editor

5.8 Cykloserver

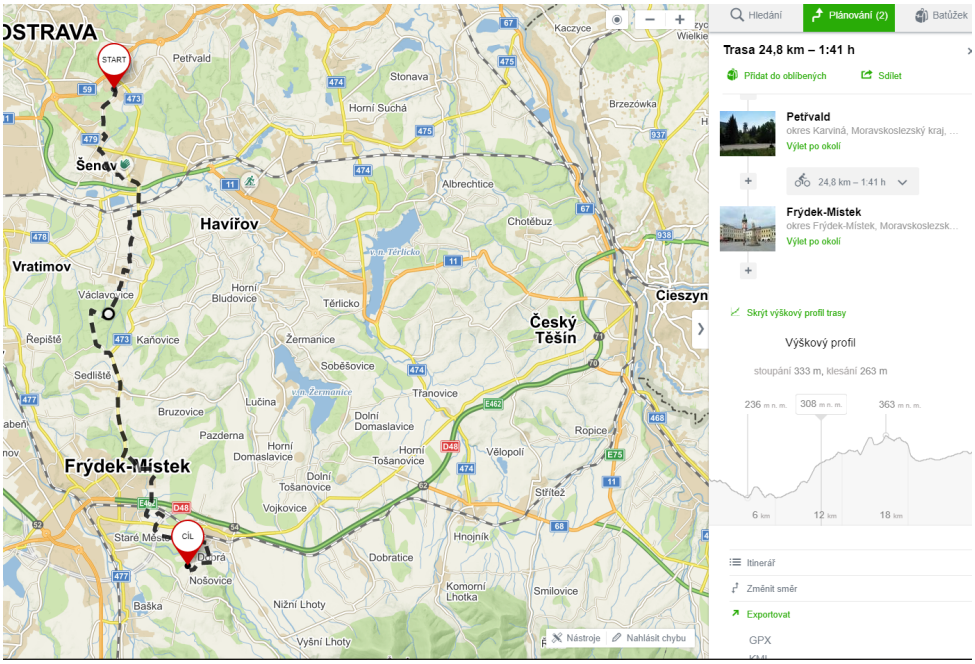
Cykloserver [82], jehož ukázku lze vidět na obrázku č. 11, je prvním českým zástupcem aplikací pro práci s track logy. Cykloserver pracuje s formátem GPX. V aplikaci lze vizualizovat pouze jednu trasu a tu je možné upravovat pomocí jednoduchého editoru. Na rozdíl od ostatních aplikací, pracuje Cykloserver s rastrovými mapami [13], proto není možné v mapách automaticky vyhledat nejkratší trasu mezi dvěma body. Trasa tak musí být naplánována manuálně bod za bodem. Při plánování trasy lze zobrazit její výškový profil. Mapy do aplikace dodává firma SHOCart [79], která se specializuje na tvorbu turistických a cyklistických map. Proto jsou mapy pro cyklistiku a turistiku velmi detailní.



Obrázek 11: Ukázka aplikace Cykloserver

5.9 Mapy.cz

Mapy.cz [83], jejichž ukázku lze vidět na obrázku č. 12, neposkytují funkcionalitu správy tras a logů GPS zařízení, ale je zde integrován velmi propracovaný plánovač tras. Systém obsahuje velmi detailní turistické, cyklistické, běžkařské a vodácké mapy, přičemž plánovač tras lze použít pro všechny způsoby dopravy. Při plánování trasy uživatel určí začátek a konec trasy a aplikace již automaticky vyhledá nejkratší cestu pro zvolený dopravní prostředek, přičemž lze zadávat další body, kterými bude trasa procházet. Při plánování trasy je možné zobrazit výškový profil trasy. Naplánovanou trasu lze exportovat pouze do GPX souboru.



Obrázek 12: Ukázka aplikace Mapy.cz

5.10 Shrnutí

Komerční systémy jsou obvykle velmi komplexní a mnohdy jsou provázány se zařízeními, které vyrábí stejná firma. Proto je často nelehké spravovat track logy vytvořené na zařízení jiných společností. Aplikace, které nejsou podporovány firmami vyrábějícími GPS zařízení nebo nejsou dotovány jiným způsobem, mají ve většině případů omezenou funkcionalitu a v mnoha případech obsahují chyby. Dalším nedostatkem výše uvedených aplikací je nemožnost vizualizace většího množství tras současně.

6 Výběr technologií

Mnoho vývojářů je při vývoji aplikací značně neefektivní z důvodu znalosti malého počtu použitelných frameworků a technologií. Cílem této kapitoly je porovnat nejznámější databázové systémy, vývojové prostředky, technologie a frameworky a vybrat z nich ty nejvhodnější pro vytvoření informačního systému pro správu a vizualizaci logů.

6.1 Databáze systém

Relační databázové systémy poskytují velmi podobnou funkcionalitu a obvykle se liší pouze v dostupných administračních nástrojích. V dnešní době se používají nástroje objektově relačního mapování, které zajišťují nezávislost aplikace na databázovém systému. Protože předpokládám použití těchto nástrojů ve své aplikaci, nebudu se výběrem databázového systému zabývat. Aplikaci budu vyvíjet na MS SQL 2016 [74], přičemž při nasazeni aplikace do ostrého provozu bude možno velmi jednoduše přizpůsobit aplikaci jinému databázovému systému, který bude dostupný na zvoleném aplikačním serveru.

6.2 Objektově Relační Mapování

Objektově Relační Mapování neboli ORM [4] je technika, která poskytuje konverzi dat mezi relační databází a objektově orientovaným programovacím jazykem. Jejím základem je doménový model, kde třídy odpovídají tabulkám v databázi. Každý objekt je pak zodpovědný za ukládání dat do databáze a jejich načítání z ní. Dále je každý objekt zodpovědný za dodržování doménové logiky aplikované na samotná data. Tímto přístupem zajistíme typovou bezpečnost datového modelu a zároveň nezávislost na databázovém serveru. V dnešní době existuje celá řada ORM frameworků, přičemž mezi ty nejznámější patří Eloquent [16], Hibernate [14] a Entity Framework [58].

6.2.1 Eloquent

Eloquent [16] byl vyvinut komunitou zabývající se vývojem frameworku Laravel [16]. Výhodou Eloquentu je minimální konfigurace a snadná integrace do Laravelu. Za nevýhodu považuji samotný programovací jazyk PHP [41], který je při mapování používán. Tento jazyk je dynamicky typovaný, což znamená, že typ proměnné je vázán na hodnotu, nikoli na proměnnou. Z toho důvodu je poměrně složité zajistit typovou kontrolu nad doménovým modelem. Dalším velkým mínusem je nemožnost vytvoření migračních skriptů z doménového modelu. Všechny skripty musí programátor vytvořit manuálně, což v kombinaci s dynamickým typováním vede k velkému množství chyb.

6.2.2 Hibernate

Hibernate [14] mapuje java objekty na entity v relační databázi. K tomu používá tzv. mapovací soubory, ve kterých je popsáno, jakým způsobem se mají data z objektu transformovat do databáze a jakým způsobem se z databázových tabulek mají vytvořit objekty. Výhodou frameworku je dotazovací jazyk HQL (Hibernate Query Language) [14], který je odvozen z SQL [2] a je mu tedy velice podobný. Další výhodou je typová kontrola nad datovým modelem. Nevýhodou je nemožnost vytvoření migračních skriptů.

6.2.3 Entity Framework

Entity Framework [58] je ORM společnosti Microsoft, který pracuje s ADO.NET [5]. Stejně jako u předchozích ORM i Entity Framework je nezávislý na databázi a programátor tak nemusí znát syntaxi jednotlivých databázových systémů. Jsou zde nástroje pro generování migračních skriptů z datového modelu, nebo také nástroje pro generování datového modelu z databáze. Vývojář si může velice snadno pomocí nástroje reverzního inženýrství vygenerovat mapu databáze, včetně všech vztahů mezi tabulkami a to je jen malé množství výhod tohoto frameworku. Samozřejmostí je také typová kontrola podobně jako tomu bylo u Hibernate. Za velkou výhodu tohoto frameworku považuji možnost využívání dotazovacího jazyka LINQ to SQL [7]. Díky LINQ to SQL lze konstruovat velmi rozsáhlé dotazy, podobně jako v SQL a přitom je zachována typová kontrola a nezávislost na použité databázi. Dalším kladem je možnost výběru mezi automatickými nebo manuálními migracemi datové vrstvy.

Pokud možnosti Entity Frameworku nejsou dostačující, existuje rozšíření Entity Framework Extensions [59], které umožňuje práci s kolekcemi bez použití speciálních funkcí SŘDB. Je-li toto rozšíření použito, v aplikaci zůstane zachována typová kontrola, kód je přehlednější a aplikace zůstává nezávislá na použitém databázovém systému.

6.3 Aplikační vrstva

Aplikační vrstva (označovaná také backend) je část aplikace spuštěna na straně serveru. Nejefektivnějším a nejpoužívanějším způsobem tvorby aplikační vrstvy se v posledních letech stala architektura REST (z anglického Repesentation State Transfer) [4]. Je to způsob, jak jednoduše číst, editovat nebo mazat informace pomocí HTTP volání [43]. Tímto přístupem lze vytvořit doménovou logiku celé aplikace na straně serveru a aplikacím pak nabídnout přístup ke stavům aplikace nebo metody pro změnu stavu aplikace. Je dostupných mnoho frameworků pracujících s REST architekturou. Mezi nejznámější patří Lumen [17], JAX-RS [18], Firebase [45] a Microsoft ASP.NET WebAPI [6].

6.3.1 Lumen

Lumen [17] se řadí mezi nejlepší PHP frameworky vhodné k tvorbě REST architektury. Tento framework je derivací frameworku Laravel [16]. Jako ORM lze použít Eloquent a k autentizaci a autorizaci Auth0 [46]. Výhodou frameworku je jeho jednoduchost a snadné nasazení hotové aplikace. Za kladnou i zápornou stránku můžeme považovat dynamické typování. Za výhodu i nevýhodu můžeme považovat automatické mapování objektů do JSON objektů. Tato funkcionalita usnadňuje práci při vývoji jednoduchých API, ale v případě rozsáhlejší aplikace můžeme tuto vlastnost považovat za bezpečnostní riziko.

6.3.2 JAX-RS

JAX-RS [18] zprostředkovává funkcionalitu pro snadné vytváření REST architektury na libovolné platformě napsané v jazyce Java. JAX-RS využívá anotace, kterými lze specifikovat chování metod třídy. Všechna data procházející těmito metodami nemusí být mapovány do DTO, což lze stejně jako v případě frameworku Lumen považovat za nedostatek i výhodu. Pro autorizaci a autentizaci uživatelů lze rovněž využít například službu Auth0, či můžeme využít zabezpečení integrované v aplikačním serveru. Implementace ovšem není triviální a mnohdy ji lze využít pouze na komerčních aplikačních serverech jako je například Glassfish [72] nebo Jboss [73].

6.3.3 Microsoft ASP.NET WebAPI

Microsoft ASP.NET WebAPI [45] společnosti Microsoft přináší rovněž jako JAX-RS funkcionalitu k snadnému vytváření REST a Micro service architektury. Podporované jazyky jsou potom C#, C++ a Visual Basic. Podobně jako u JAX-RS jsou zde využívány anotace metod, kterými lze specifikovat, jak se má anotovaná metoda chovat. Chování je potom velmi podobné frameworku JAX-RS, přičemž musíme objekty mapovat do DTO. Stejně jako v předchozích technologiích, tuto funkcionalitu můžeme považovat jako nedostatek. Další výhodou je anotace Authorize [6], která může být vázána na metody nebo celé třídy, a lze s ní specifikovat povolenou funkcionalitu jednotlivých rolí v systému.

6.3.4 Firebase

Platforma Firebase [45] poskytuje velké množství funkcí, díky kterým lze velmi snadno vytvořit aplikační vrstvu bez potřeby znalosti rozsáhlých frameworků, způsobu jejich nasazení a implementace zabezpečení. Firebase obsahuje funkci autorizace a autentizace uživatelů, uložiště, realtime databázi a dále možnost spouštět javascriptový kód na serveru. Pro každou funkci jsou k dispozici knihovny pro nejrozšířenější frameworky díky čemu je implementace velmi snadná.

6.4 Prezentační vrstva

Prezentační vrstva (označovaná také frontend) je část aplikace, se kterou pracuje uživatel. Po desítkách letech vývoje nástrojů pro práci s prezentační vrstvou, se tyto nástroje rozdělili na nástroje pro vývoj logiky prezentační vrstvy a nástroje usnadňující tvorbu prvků uživatelského rozhraní. V této kapitole jsou stručně popsány nástroje Jquery [15], Angular JS [22], Angular 2+ [48] a Vue JS [44] sloužící k vývoji logiky prezentační vrstvy a Material Design Boostrap [27] a Telerik UI for ASP.NET MVC [52] určené k tvorbě uživatelského rozhraní.

6.4.1 Jquery

Jquery [15] je jeden z nejstarších javascriptových frameworků, jehož první verze byla vydána v roce 2006. Filozofie tohoto frameworku je oddělení struktury HTML a chování uživatelského rozhraní. To v praxi znamená, že vývojář najde pomocí Jquery element, který slouží například jako tlačítko a místo definice události on-click, změní jeho manipulátor události. Tento přístup se nazývá principem nevtíravého javascriptu. S příchodem SPA (z anglického Single Page Application) [22] aplikací se ovšem tento přístup stal velmi neefektivní, neboť neumožňuje snadno odchytávat větší množství událostí a proto bylo zapotřebí najít efektivnější přístup k tvorbě webových a speciálních desktopových aplikací.

6.4.2 Angular JS

Angular JS [22] byl speciálně navržen pro tvorbu SPA aplikací. Využívá k tomu návrhový vzor Model View Whatever [22]. Novinkou byl takzvaný Two way databinding [22], který tento framework vynesl mezi nejpopulárnější, ale zároveň se velmi brzy objevily nedostatky tohoto přístupu a framework ztratil svou popularitu. Funkcionalitu Two way databinding obstarává vlastní implementace Scope [22] v AngularJS. Scope je proměnná definovaná v kontroléru aplikace a jsou v ní uloženy data aplikace, přičemž lze tyto data pomocí speciálních direktiv velmi snadno prezentovat v uživatelském rozhraní a zároveň při změně proměnné uživatelem, je tato změna projektována do modelu aplikace nebo obráceně. Velké množství komponent v aplikaci a velké množství dat ve scopu ovšem zabírá mnoho paměti, způsobuje vysokou latenci uživatelského rozhraní, případně i pád aplikace.

6.4.3 Angular 2+

Angular 2+ [48] odstraňuje nedostatky AngularJS a přidává podporu typové kontroly pomocí Typescriptu [49]. Jedná se o společný projekt společností Microsoft a Google. Struktura aplikace je poměrně složitá, proto je pro Angular 2+ k dispozici CLI, který umožňuje poměrně snadno založení projektu, generování komponent, testování nebo formátování kódu. Při vývoji architektury Angularu 2+ byl použit komponentový model a všechny komponenty jsou optimalizovány pro web, mobilní web a nativní aplikace. Tento framework je dostupný zdarma pod MIT licencí.

K efektivní tvorbě aplikací je ovšem vhodné použít komponenty s integrovanými UI elementy, které vývojáři usnadní mnoho práce, protože je lze velmi snadno integrovat do doménového modelu aplikace. Tyto komponenty už bohužel nebývají zdarma a za ty kvalitní může vývojář zaplatit desítky tisíc korun.

6.4.4 Vue JS

Vue JS [44] je další opensource framework, který vznikl kombinací vlastností frameworku AngularJS a React [85]. Tento framework byl vytvořen jako odlehčená knihovna Angular JS bez složitých konceptů. S novými verzemi obou frameworků můžeme sledovat jejich velkou podobnost. Je zřejmé, že se tvůrci vzájemně inspirují ve zdokonalování svého softwaru. VueJS je možné integrovat do libovolné webové aplikace. Bohužel zde není zajištěna typová kontrola, jako je tomu u Angularu 2+.

6.4.5 Nástroje usnadňující tvorbu uživatelského rozhraní

Material Design Boostrap [27] je sada CSS tříd a komponent pro vývoj uživatelského rozhraní. Tyto styly je možné integrovat do všech výše uvedených frameworků. Komponenty jsou bohužel odladěné pouze na statických stránkách a při kombinaci s výše uvedenými frontendovými nástroji, narazí vývojář na řadu nedostatků. Například komponenty Dropdownlist nebo Menu nefungovaly ani v případě použití referenčního kódu z oficiálního webu. Cena komponent byla v době psaní této diplomové práce poměrně nízká (79 EUR), což odpovídalo kvalitě softwaru.

Telerik UI for ASP.NET MVC [52] považuji za nejlepší nástroj pro vývoj uživatelského rozhraní. Ke každé komponentě jsou k dispozici detailní ukázkové kódy a dokumentace. Pro Visual Studio je dostupné rozšíření, které usnadňuje konverzi stávajících projektů na Telerik projekty, kde jsou integrovány knihovny pro práci s komponentami. Další výhodou je výběr vzhledu aplikace jedním klikem, přičemž lze vybírat z nejmodernějších vzhledů, jako je například Material Design. Další výhodou je integrovaná funkcionalita lazy loadingu [4] a websocketu [51] ve všech komponentách, které mohou pracovat s objemnějšími daty. Telerik UI for ASP.NET MVC je integrován do Razor enginu [6], díky čemu je v komponentách integrována typová kontrola. Cena komponent je vysoká, v době psaní diplomové práce to bylo $999 za licenci pro jednoho vývojáře. Vzhledem k efektivitě práce s komponentami se ale tato investice velice rychle vrátí.

6.5 Shrnutí

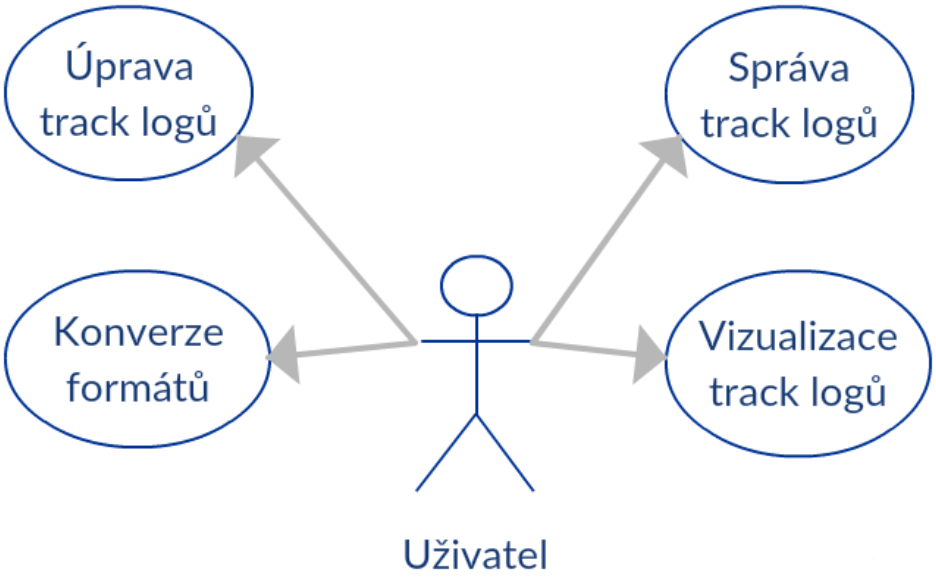
Výběr technologií není vždy jednoznačný. Mnohdy si investor nebo vývojáři neuvědomí, jakého rozsahu může aplikace nabýt po letech vývoje, a proto je použit nevhodný framework, který například nezajišťuje typovou kontrolu. Jindy se zvolí zbytečně rozsáhlý a komplexní framework, který svou složitostí zpomaluje vývoj. Často jsou vyvíjena vlastní řešení, která se ovšem vyplácí převážně velkým firmám, pro které je výhodnější zaměstnat větší množství vývojářů místo platby

velkých finančních částek za komerční produkty. V minulosti jsem měl možnost pracovat se všemi uvedenými nástroji. Telerik UI for ASP.NET MVC v kombinaci s Entity Frameworkem a Entity Framework Extensions považuji za nejspolehlivější a nejefektivnější nástroje. Za velkou výhodu považuji typovou kontrolu v aplikační vrstvě a zároveň ve vrstvě prezentační. Dalším kladem je stabilita všech vybraných technologií na rozdíl od opensource technologií, jež často mění logiku frameworku. Tím mnohdy nejsou zpětně kompatibilní a neexistují u nich migrační nástroje. Bez těchto výhod se efektivita vývoje aplikace velmi rapidně snižuje. Z těchto důvodů jsem pro tento projekt vybral nástroje Telerik UI for ASP.NET MVC, Entity Framework a Entity Framework Extensions.

7 Analýza

7.1 Vize

Cílem této práce je vytvořit informační systém pro správu tras a logů z GPS zařízení, zpracovávajících pozici objektů a prezentaci pozičních údajů na mapových podkladech. Systém bude dále obsahovat funkce úpravy pozičních dat, tedy jejich konverze, spojování, ořezání a rozdělování. Data bude nutné prezentovat vhodným způsobem, aby nedocházelo ke špatné orientaci mezi trasami. Bude se jednat o webovou aplikaci s responsivním designem. Uživatel si v aplikaci vytvoří uživatelský účet, který bude umožňovat správu logů uživatele, jejich vizualizaci a sdílení. Diagram užití lze nalézt na obrázku č. 13



Obrázek 13: Diagram užití

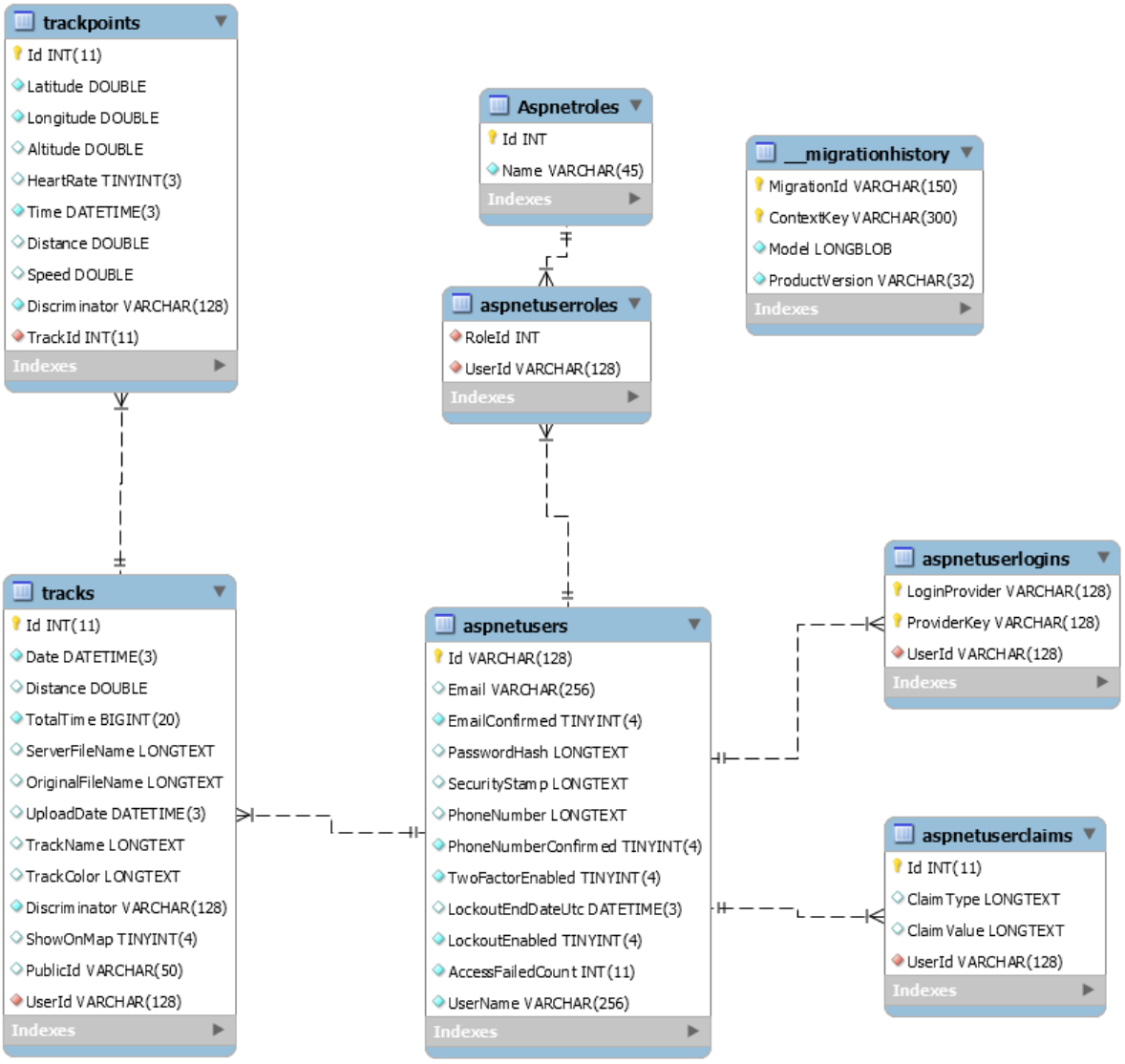
7.2 Datová analýza

Vstupy do informačního systému rozumíme informace, které budou systémem zpracovány a které budeme shromažďovat v databázi. S těmito údaji budu dále v systému pracovat a rozvíjet je v dalších funkcích.

- Uživatel – Identifikace uživatele, uživatelské jméno, email a hash hesla. Dále zde bude prostor pro rozšíření o dvoufázovou autentizaci, bezpečnostní razítko využívané pro reset hesla, potvrzení emailu, potvrzení telefonního čísla a uzamčení uživatelského účtu.
- Uživatelská role – Název role. Bude využíván pro rozdělení rolí uživatelů.

- Trasa – Základní informace o trase. Název trasy, délka trasy v kilometrech, celkový čas, barva vizualizace v mapových podkladech, název pracovního souboru, původní název souboru a informace, zda se má trasa zobrazit na mapě
- Bod trasy – Informace o pozici objektu v čase. K uložení bodů bude využit souřadnicový systém WGS84. Rychlost přesunu a vzdálenost mezi předchozím a aktuálním bodem a aktuální srdeční tep, bude-li podporován logovacím zařízením.

Na obrázku č. 14 se nachází ER model informačního systému. Datový slovník je k dispozici v tabulkách č. 1, 2, 3, 4, 5, 6, 7



Obrázek 14: ER diagram

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
Id	nvarchar	128	ANO	NE	ANO	Identifikátor tabulky
Email	nvarchar	256	NE	ANO	NE	Email uživatele
EmailConfirmed	bit		NE	NE	NE	Příznak potvrzeného emailu
PasswordHash	nvarchar	MAX	NE	ANO	NE	Hash hesla
SecurityStamp	nvarchar	MAX	NE	ANO	NE	Bezpečnosti razítko pro reset hesla.
PhoneNumber	nvarchar	MAX	NE	ANO	NE	Telefonní číslo
PhoneNumber Confirmed	bit		NE	NE	NE	Příznak potvrzeného Telefonní číslo
TwoFactor Enabled	bit		NE	ANO	NE	Příznak použití dvoufázového přihlašování
LockoutEnd DateUtc	datetime		NE	ANO	NE	Datum a čas uzamčení účtu
LockoutEnabled DateUtc	bit		NE	NE	NE	Příznak uzamčení účtu
AccessFailed Count	int		NE	NE	NE	Počet špatných přihlášení
UserName	varchar	256	NE	NE	NE	Uživatelské jméno

Tabulka 1: Schéma tabulky AspNetUsers

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
Id	nvarchar	128	ANO	NE	ANO	Identifikátor tabulky
Name	nvarchar	45	NE	NE	NE	Název role

Tabulka 2: Schéma tabulky AspNetRoles

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
UserId	nvarchar	128	ANO	NE	ANO	Identifikátor uživatele
RoleId	nvarchar	128	ANO	NE	ANO	Identifikátor role

Tabulka 3: Schéma tabulky AspNetUsersRoles

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
LoginProvider	nvarchar	128	ANO	NE	ANO	Poskytovatel autentizace
ProviderKey	nvarchar	128	ANO	NE	ANO	Identifikátor uživatele dodaný poskytovatelem autentizace
UserId	nvarchar	128	ANO	NE	ANO	Identifikátor uživatele

Tabulka 4: Schéma tabulky AspNetUsersLogins

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
Id	int		ANO	NE	ANO	Identifikátor záznamu
Date	datetime		NE	NE	NE	Datum vytvoření trasy
Distance	float		NE	ANO	NE	Celková délka trasy
TotalTime	bigint		NE	NE	NE	Celkový čas trasy
UserId	nvarchar	128	ANO	NE	ANO	Identifikátor uživatele
ServerFile Name	nvarchar	MAX	NE	ANO	NE	Název souboru na serveru
OriginalFile Name	nvarchar	MAX	NE	ANO	NE	Název původního souboru
UploadDate	datetime		NE	ANO	NE	Datum a čas nahrání trasy
TrackName	nvarchar	128	NE	ANO	NE	Název trasy
TrackColor	nvarchar	10	NE	ANO	NE	Barva trasy
Discriminator	nvarchar	128	NE	NE	NE	Využívá EF k mapování polymorfních objektů
ShowOnMap	bit		NE	ANO	NE	Příznak zda se bude trasa zobrazovat v mapě
PublicId	nvarchar	50	NE	ANO	ANO	Veřejný klič pro zobrazení trasy

Tabulka 5: Schéma tabulky Track

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
Id	int		ANO	NE	ANO	Identifikátor záznamu
TrackId	int		ANO	NE	ANO	Identifikátor trasy
Latitude	float		NE	NE	NE	Zeměpisná šířka
Longtitude	float		NE	NE	NE	Zeměpisná délka
Altitude	float		NE	NE	NE	Nadmořská výška
HeartRate	tinyint		NE	ANO	NE	Okamžitý srdeční tep
Time	datetime		NE	NE	NE	Čas zaznamenání bodu trasy
Distance	float		NE	ANO	NE	Vzdálenost od předchozího bodu
Speed	float		NE	ANO	NE	Okamžité rychlost v bodě
Discriminator	nvarchar	128	NE	NE	NE	Využívá EF k Mapování polymorfních objektů

Tabulka 6: Schéma tabulky TrackPoints

Atribut	*Dat.Typ*	*Velikost*	*PK*	*Null*	*Index*	Popis atributu
Id	int		ANO	NE	ANO	Identifikátor záznamu
UserId	nvarchar	128	ANO	NE	ANO	Identifikátor uživatele
ClaimType	longtext		NE	NE	NE	ASP.NET Identity
ClaimValue	longtext		NE	NE	NE	ASP.NET Identity

Tabulka 7: Schéma tabulky AspNetUsersClaims

7.3 Funkční analýza

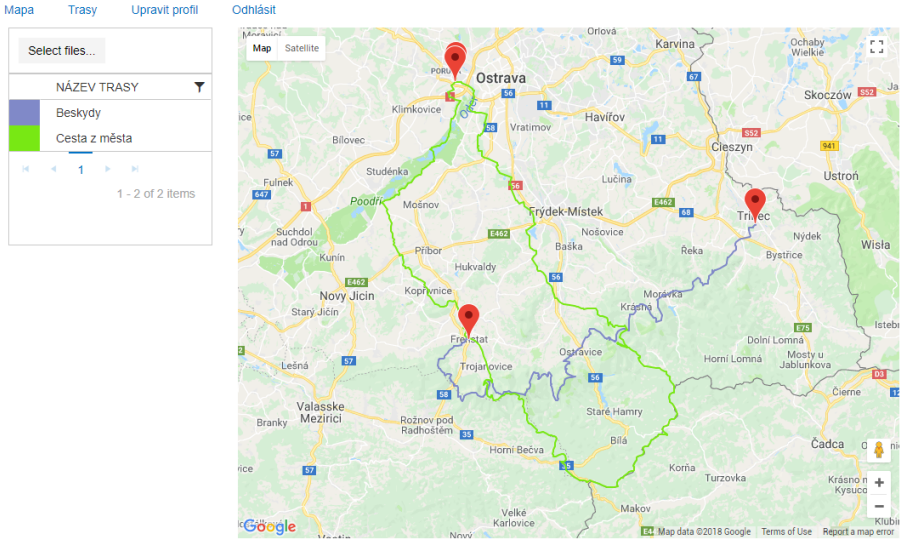
- Správa track logů
  - Úprava názvu trasy
  - Úprava vizualizace trasy
  - Odstranění vadných dat z track logu
- Úprava track logů
  - Rozdělení trasy
  - Spojení tras
  - Ořezání trasy
- Konverze formátů
  - Import track logů
  - Export track logů
- Vizualizace track logů
  - Vizualizace trasy
  - Vizualizace rychlostí

8 Implementace

8.1 Základní popis systému

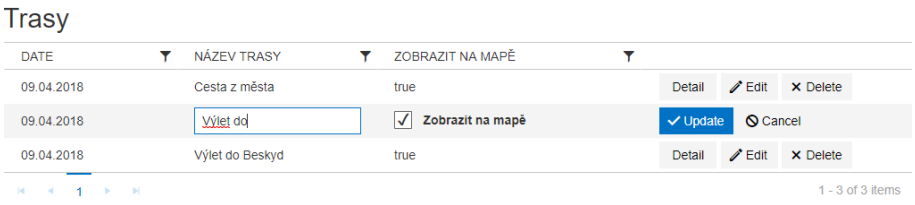
Aplikace se skládá z menu, které se mění v závislosti jestli je uživatel přihlášen. Je-li uživatel přihlášen, lze přepínat mezi zobrazením mapy, správou tras a úpravou profilu. Pokud uživatel není přihlášen, má možnost registrace nebo přihlášení.

- Záložka Mapa je rozdělena na dvě části. Levá část obsahuje komponentu pro nahrávání nových souborů a komponentu zobrazující seznam tras, které jsou vykresleny na mapě. V pravé časti je vykreslena mapa s aktivními trasami. Ukázku lze vidět na obrázku č. 15.



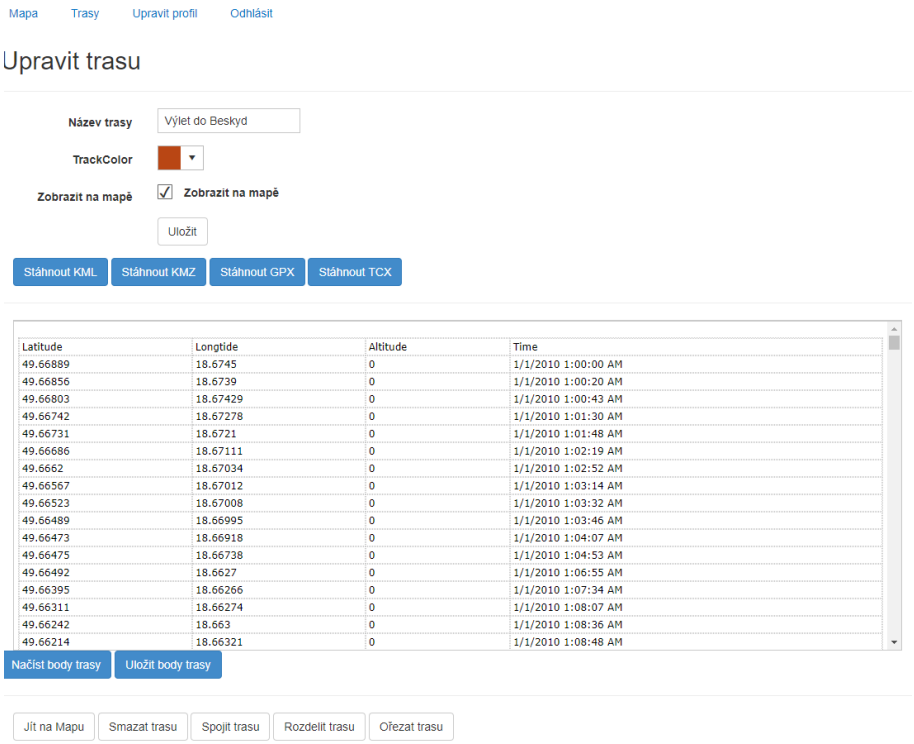
Obrázek 15: Ukázka zobrazení mapových podkladů

- V záložce Trasy je seznam všech tras uživatele, jehož ukázku lze vidět na obrázku č. 16. V této komponentě lze editovat některá metadata trasy. Jsou podporovány funkce třídění a řazení položek podle libovolného atributu.



Obrázek 16: Ukázka seznamu tras

Editaci trasy lze provádět po kliknutí na tlačítko *Detail*. Jak lze vidět na obrázku č. 17, na této stránce lze provádět úpravu názvu trasy, barvu vizualizace v mapě, editaci bodů trasy, spojení s jinou trasou, rozdělení trasy nebo ořezání trasy. Dále lze stáhnout track log trasy v libovolném podporovaném formátu.

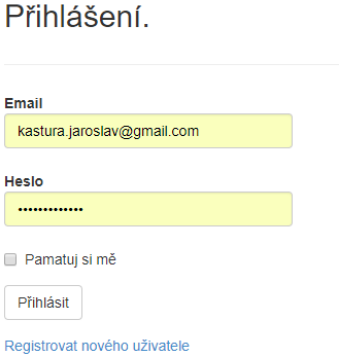


Obrázek 17: Ukázka editace trasy

- V záložce Upravit profil lze změnit uživatelské heslo.

8.2 Autentizace a autorizace

K zabezpečení aplikace jsem se rozhodl použít ASP.NET Identity [78] framework, který je součástí ASP.NET MVC a podporuje standard OAuth2 [84]. Přihlašovací a registrační formuláře jsou dostupné přes horní menu aplikace. Ukázku přihlašovacího formuláře lze vidět na obrázku č. 18. Uživatel se do aplikace přihlašuje pomocí emailové adresy a hesla.



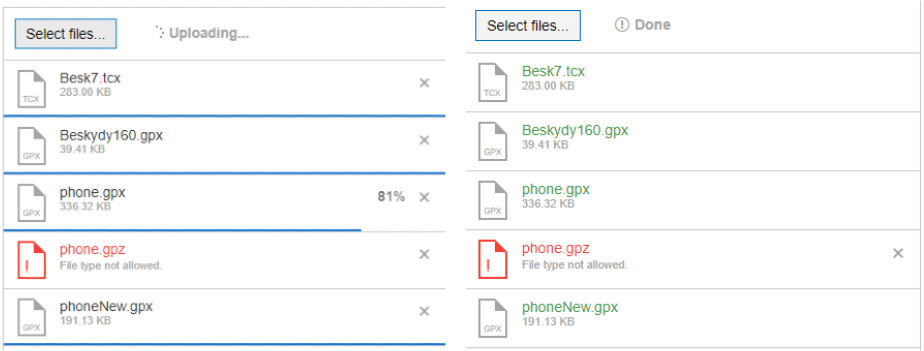
Obrázek 18: Ukázka přihlašovacího formuláře

8.3 Práce s track logy

K nahrání track logů na server používám komponentu Kendo Upload [52]. Ukázku implementace lze nalézt ve výpisu kódu č. 11 a ukázku komponenty lze vidět na obázku č. 19. Tato komponenta zajišťuje asynchronní přenos souborů mezi prezentační a aplikační vrstvou. Ve vlastnosti komponenty `Validation` jsou definovány typy souborů, které lze na server odeslat. Vlastnost `Async` nastaví komponentu do asynchronního režimu komunikace a v parametru je předán název kontroleru a metody `Save("Save", "Map")`, která obslouží POST volání komponenty Kendo Upload. Událost `Events(e => e.Complete("refresh"))` zajistí spuštění funkce `refresh`, která překreslí mapu po nahrání souborů na server. Komponenta umožňuje přenos více souboru současně.

```
@(Html.Kendo().Upload()
   .Name("files")
   .HtmlAttributes(new {aria_label = "files"})
   .Async(a => a
      .Save("Save", "Map")
      .AutoUpload(true)
   ).Events(e => e.Complete("refresh"))
   .Validation(validation => validation.AllowedExtensions(new string[] {".kml", ".kmz", ".gpx",
       ".tcx", "json"}))
)
```

Výpis 11: Ukázka nahrání trasy na server pomocí komponenty Kendo Upload



Obrázek 19: Ukázka komponenty Kendo Upload

K importu a exportu souborů jsem využil hotové opensource knihovny pro .NET[5]. První knihovnou je Math Matthey Tools TrackReaders [54]. Tuto knihovnu využívám k načítání TCX, GPX, KML a KMZ souborů a následné konverzi do vlastního formátu. Knihovna obsahuje všechny potřebné funkce pro načítání track logů. Ukázku deserializace souboru pomocí knihovny Math Matthey Tools TrackReaders lze vidět ve výpisu kódu č. 12. Pro export trasy do KML a KMZ formátu využívám knihovnu SharpKml[55], což je implementace OGC KML2.2 standardu [28]. K exportu do formátu TCX využívám knihovnu TcxTools [56], která obsahuje implementaci

všech metod potřebných k práci s tímto schématem. K exportu do formátu GPX využívám knihovnu GPXLib [86]. Ukázku serializace souboru pomocí knihovny GPXLib lze vidět ve výpisu kódu č. 13. Ukázku serializace souboru pomocí knihovny TcxTools lze vidět ve výpisu kódu č. 14.

```
Track track = Math.Tools.TrackReaders.Deserializer.File(filename);
```

Výpis 12: Ukázku deserializace souboru pomocí knihovny Math Matthey Tools TrackReaders

```
GPXLib gpx = new GPXLib
{
   Creator = "Jaroslav Kastura"
};
foreach (Model.TrackPoint trackTrackPoint in track.TrackPoints)
{
   var tmp = new Wpt
   {
      Lat = (decimal) trackTrackPoint.Latitude,
      Lon = (decimal) trackTrackPoint.Longitude,
      Time = trackTrackPoint.Time
   };
   if (trackTrackPoint.Altitude != null)
   {
      tmp.Ele = (decimal) trackTrackPoint.Altitude;
   }
   gpx.AddTrackPoint("track", 0, tmp);
}

gpx.Metadata = new Metadata
{
   Author = new Person
   {
      Email = new Email
      {
         Domain = "gmail.com",
         Id = "kastura.jaroslav"
      },
      Link = new Link
      {
         Href = "www.gipsy.cz",
         Text = "Nejaky text",
         Type = "Typ"
      }
   },
   Time = DateTime.Now,
   Bounds = new Bounds
   {
      Maxlat = (decimal) track.TrackPoints.Max(m => m.Latitude),
      Minlat = (decimal) track.TrackPoints.Min(m => m.Latitude),
      Maxlon = (decimal) track.TrackPoints.Max(m => m.Longitude),
```

```
      Minlon = (decimal) track.TrackPoints.Max(m => m.Longitude),
   },
   Copyright = new Copyright
   {
      Author = "Kastura Jaroslav",
      License = "Kastan Soft",
      Year = "2017"
   },
   Desc = "Track info generated by Gipsy",
};
gpx.SaveToFile(filename);
```

Výpis 13: Ukázka serializace GPX souboru pomocí knihovny GPXLib

```
TrainingCenterDatabase training = new TrainingCenterDatabase();
Activity activity = new Activity();
activity.Id = DateTime.Now;
activity.Sport = Sport.Biking;
activity.Creator = new Application() { Name = "Kastan" };
ActivityLap lap = new ActivityLap();
foreach (TrackPoint point in track.TrackPoints)
{
   if (point.Altitude == null)
   {
      point.Altitude = 0;
    }
   if (point.HeartRate == null) point.HeartRate = 0;
   else if (point.HeartRate > byte.MaxValue) point.HeartRate = byte.MaxValue;
   else if (point.HeartRate < byte.MaxValue) point.HeartRate = byte.MinValue;
   lap.Track.Add(new Trackpoint
   {
      Time = point.Time,
      AltitudeMeters = (double)point.Altitude,
      AltitudeMetersSpecified = true,
      Position = new Position
      {
         LatitudeDegrees = point.Latitude,
         LongitudeDegrees = point.Longitude
      },
      HeartRateBpm = new HeartRateInBeatsPerMinute()
      {
         Value = (byte)point.HeartRate
      }
   });
}
activity.Lap.Add(lap);
training.Activities.Activity.Add(activity);
XmlSerializer serializer = new XmlSerializer(typeof(TrainingCenterDatabase));
Stream writer = new FileStream(filename, FileMode.Create);
```

```
serializer.Serialize(writer, training);
writer.Close();
```

Výpis 14: Ukázka serializace TCX souboru pomocí knihovny TcxTools

Pro snadnou implementaci a rozšiřitelnost je k ukládání tras využit relační databázový systém. Metadata o trase jsou uložena v tabulce `Tracks` (Datová analýza - tabulka č. 5) a jednotlivé body trasy jsou uloženy v tabulce `TrackPoints` (Datová analýza - tabulka č. 6). Do těchto tabulek lze ukládat všechna potřebná data o trase a všech jejich bodech. Takto vytvořená datová struktura je velmi snadno rozšiřitelná a minimalistická. Další výhodou tohoto přístupu je jednoduchost implementace, protože lze všechny operace nad daty provádět prostřednictvím Entity Frameworku a Entity Framework Extensions. Jednoduchost implementace lze vidět ve výpisu kódu č. 15, který v aplikaci zajišťuje uložení trasy do databáze.

```
private void SaveTrackPoints(ICollection<TrackPointExt> trackPointExts)
{
   db.BulkInsert(trackPointExts);
}
```

Výpis 15: Uložení trasy pomocí Entity Framework Extensions

8.4 Výpočet vzdálenosti mezi body

Abych mohl zjistit rychlost přesunu objektu mezi dvěma body, je nutné vypočítat vzdálenost mezi těmito body. K tomu účelu jsem vytvořil statickou funkci `DistanceBetweenPlaces`, kterou lze vidět ve výpisu kódu č. 16. Tato funkce vypočítá ze souřadnic dvou bodů vzdálenost mezi těmito body. Pro zpřesnění výpočtu lze zadat nepovinný parametr alt, reprezentující nadmořskou výšku v metrech. V algoritmu je definována proměnná `R`, která představuje poloměr země v kilometrech. Je-li zadán nepovinný parametr `alt`, přičte se tato hodnota po převedení na kilometry `(alt / 1000)` k poloměru země. V další části je implementace The Haversine Formula [75], jenž vypočítá vzdálenost mezi dvěma body na povrchu koule. The Haversine Formula pracuje s radiány, a proto je nutné provést konverzi. Protože track logy pracují s metry, je výsledná hodnota převedena na metry (`dist * 1000`).

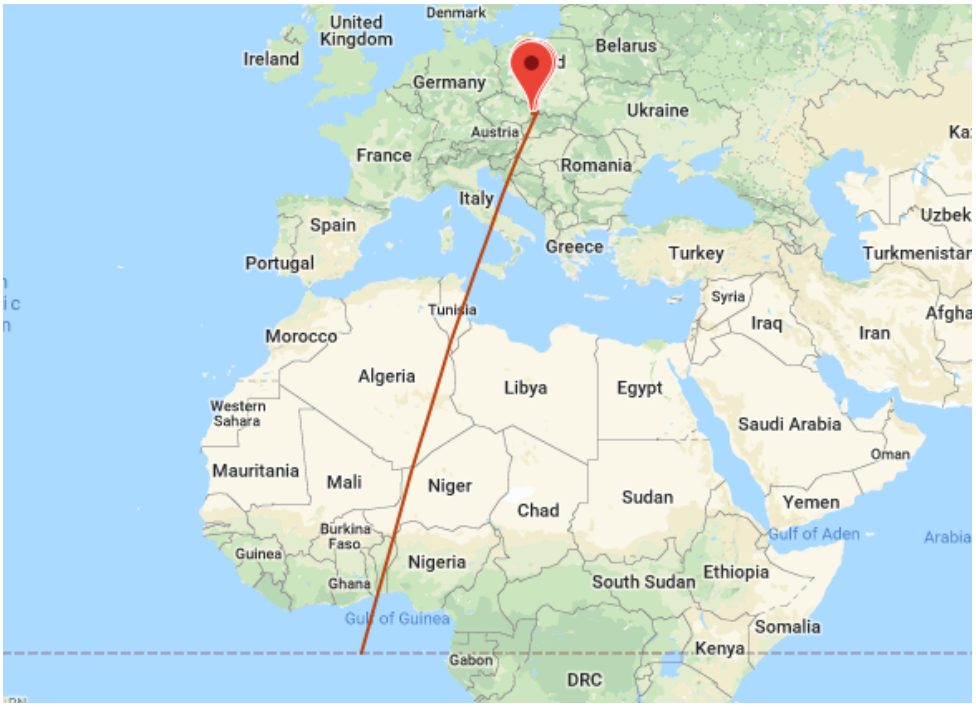
```
private static double DistanceBetweenPlaces(double lon1, double lat1, double lon2, double lat2,
    double? alt)
{
   double R = 6378;
   if (alt != null) { R += (double) (alt / 1000); }
   double sLat1 = System.Math.Sin(ConvertDegreesToRadians(lat1));
   double sLat2 = System.Math.Sin(ConvertDegreesToRadians(lat2));
   double cLat1 = System.Math.Cos(ConvertDegreesToRadians(lat1));
   double cLat2 = System.Math.Cos(ConvertDegreesToRadians(lat2));
   double cLon = System.Math.Cos(ConvertDegreesToRadians(lon1) - ConvertDegreesToRadians(lon2));
   double cosD = sLat1 * sLat2 + cLat1 * cLat2 * cLon;
```

```
    double d = System.Math.Acos(cosD);
    double dist = R * d;
    return dist * 1000;
}
private static double ConvertDegreesToRadians(double degrees)
{
    double radians = (System.Math.PI / 180) * degrees;
    return (radians);
}
```

Výpis 16: Výpočet vzdálenosti mezi body trasy

8.5 Identifikace a odstranění chybných bodů v logu

Nezřídka se stává, že se v track logu vyskytnou vadné body. Tato situace nastává například, když zařízení není schopno zpracovat lokalizační údaje z důvodu špatného pokrytí signálem v daném místě, nebo jiné chyby při zpracovaní lokalizačních dat. V takovém případě jsou v track logu data, která nekorespondují s reálnou trasou. Ukázku trasy s chybným bodem lze vidět na obrázku č. 20.



Obrázek 20: Ukázka trasy s chybným bodem

Definice 3 *Za odlehlá pozorování považujeme ty hodnoty proměnné, které se mimořádně liší od ostatních hodnot a tím ovlivňují například vypovídající hodnotu průměru.*

Takové body lze jednoduše najít výpočtem rychlosti přesunu objektu mezi dvěma body. Pokud je rychlost přesunu větší než předem definovaná hranice maximální rychlosti, můžeme body označit za vadné a odstranit je z track logu. Ovšem toto řešení je příliš krátkozraké, neboť předpokládáme využití tohoto IS pro track logy z různých objektů. Pokud bychom toto pravidlo aplikovali například pro letadlo, přičemž bychom předpokládali využití systému maximálně pro automobil, mohli by být z track logu odstraněny všechny body trasy. Proto jsem využil znalostí, které jsem získal v kurzu statistiky a na tuto problematiku použil statistické metody k identifikaci odlehlých pozorování pomocí vnitřních hradeb [3]. Při testování jsem dosáhl 100% úspěšnosti ve vyhledávání chybných lokalizačních dat, přičemž implementace algoritmu byla ze všech statistických metod pro vyhledání odlehlých pozorování nejefektivnější a nejjednodušší.

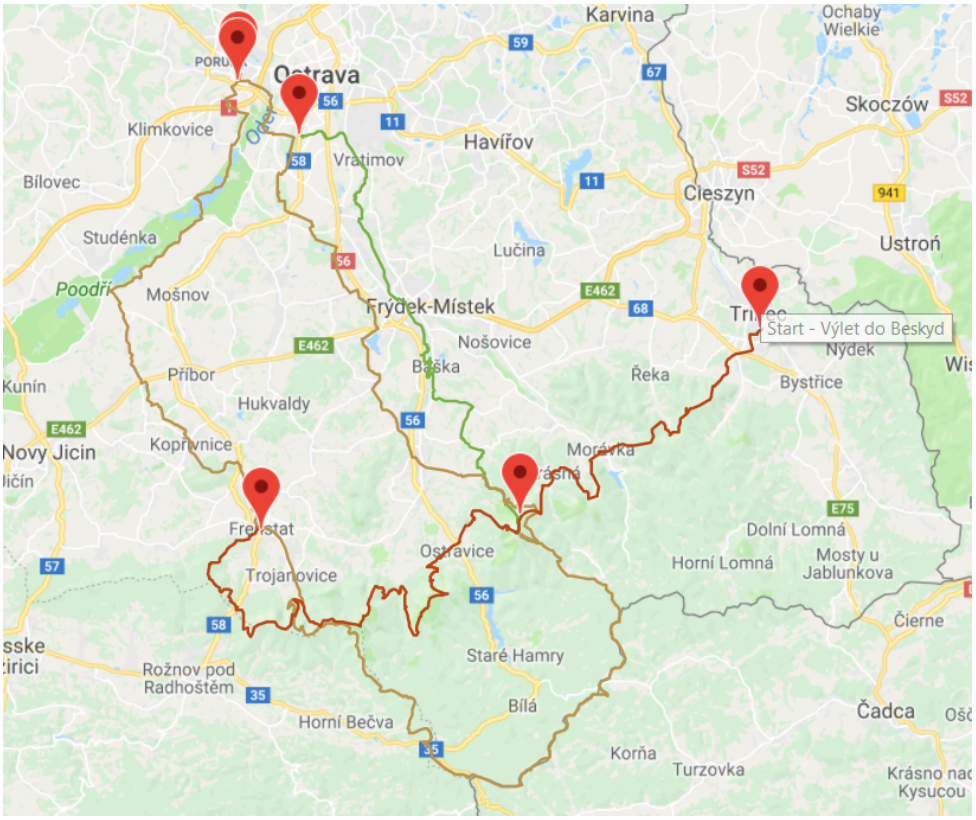
Ve výpisu kódu č. 17, lze vidět implementaci algoritmu k identifikaci odlehlých pozorování pomocí vnitřních hradeb. V tomto algoritmu jsou všechny body trasy rozděleny do čtyř kvartilů. Následně je z prvního kvartilu určena spodní hranice oblasti (`downBorder`) a z třetího kvartilu je určena horní hranice oblasti (`upBorder`). Z těchto dvou bodů je v další části algoritmu vypočítáno Interkvartilové rozpětí `IRQ` a horní hranice maximální rychlosti upExtreme (`upExtreme`), jenž představuje maximální povolenou rychlost mezi dvěma body. Pokud je mezi dvěma body hodnota `TrackPoint.Speed` vyšší než hodnota `upExtreme`, musí být tento bod odstraněn z kolekce. Při odstranění bodu z kolekce je nejdříve nutné vytvořit pomocnou kolekci chybných bodů. Do této kolekce jsou pomoci LINQ načteny všechny body s hodnotou `TrackPoint.Speed` větší než je hodnota `upExtreme`. Následně jsou tyto body odstraněny z původní kolekce.

```
public static void RemoveErrorsFromTrack(Track track)
{
   ICollection<TrackPoint> trackPoints = track.TrackPoints.OrderBy(s => s.Speed).ToList();
   int kvartiles = trackPoints.Count / 4;
   TrackPoint downBorder = trackPoints.ElementAt(kvartiles * 1);
   TrackPoint upBorder = trackPoints.ElementAt(kvartiles * 3);
   double IRQ = 0;
   double downExtreme = 0;
   double upExtreme = 0;
   if (downBorder.Speed != null && upBorder.Speed != null)
   {
      IRQ = (double)upBorder.Speed - (double)downBorder.Speed;
      upExtreme = (double)upBorder.Speed + (1.5 * IRQ);
   }
   ICollection<TrackPoint> extremesPoints = trackPoints.Where(t => t.Speed > upExtreme).ToList()
       ;
   foreach (TrackPoint point in extremesPoints)
   {
      TrackPoint tmp = track.TrackPoints.First(p => p.StatisticsId == point.StatisticsId);
      track.TrackPoints.Remove(tmp);
   }
}
```

Výpis 17: Algoritmus na odstranění vadných bodu v track logu

8.6 Vizualizace tras

Vykreslení tras do mapových podkladů, které můžete vidět na obrázku č. 21, provádí po inicializaci Google Maps funkce `downloadTracks()`, kterou lze vidět ve výpisu kódu č. 19. Tato funkce nejprve načte všechny aktivní trasy, které ji poskytne metoda `GetActiveTracks()`, kterou lze vidět ve výpisu kódu č. 18. Hodnota proměnné je v dalším kroku použita Entity Frameworkem k vyhledání všech tras, které jsou označeny jako viditelné (tzn. atribut `ShowOnMap == true`). Z tohoto dotazu je vytvořen nový objekt, respektive kolekce objektů tras, které jsou serializovány do JSON objektu a odeslány klientovi.



Obrázek 21: Ukázka vizualizace tras v mapových podkladech

```
public JsonResult GetActiveTracks() {
   string userId = db.Users.Find(System.Web.HttpContext.Current.User.Identity.GetUserId()).Id;
   var tracks = db.Tracks.Where(a => a.ShowOnMap == true)
      .Where(u => u.UserId == userId)
      .Include(p => p.TrackPoints)
      .Select(t => new {
        t.Id, t.TrackName, t.TrackColor, t.TrackPoints
```

```
      }).ToList();
   return new JsonResult { Data = tracks };
}
```

Výpis 18: Příprava dat pro vykreslení tras do mapy

```
function downloadTracks() {
   $.ajax({
      url: "/api/ActiveTracks",
      success: function(result) {
         tracks = result.Data;
         map.fitBounds(getPathsBounds());
         drawPaths();
         drawChart();
      }
   });
}
```

Výpis 19: Načtení aktivních tras pomocí Ajaxu

Jakmile klientská aplikace načte kolekci tras z JSON objektu, je spuštěna funkce `getPathsBounds()`, která nalezne okrajové části mapy tak, aby byly všechny vybrané trasy viditelné na mapě. Kód funkce je uveden ve výpise č. 21. Funkce prochází všechny body ve vybraných trasách a vyhledává minimální (`latMin, lngMin`) a maximální (`latMax, lngMax`) hodnoty zeměpisné šířky a délky. Z výsledných hodnot je vytvořen objekt `google.maps.LatLngBounds`, který ohraničuje oblast, ve které se nachází všechny trasy. Ten je předán do Google Maps metodou `map.fitBounds()`.

```
function getPathsBounds() {
   var latMin = null; var lngMin = null; var latMax = null; var lngMax = null;
   var setLatMin = function(path) {
      for (var i = 0; i < path.TrackPoints.length; i++) {
         if (latMin == undefined) {
            latMin = path.TrackPoints[i].Latitude;
         } else if (latMin > path.TrackPoints[i].Latitude) {
            latMin = path.TrackPoints[i].Latitude;
         }
      }
   };
   var setLatMax = function(path) {.....};
   var setLngMin = function(path) {.....};
   var setLngMax = function(path) {.....};
   for (var i = 0; i < tracks.length; i++) {
      setLatMax(tracks[i]); setLatMin(tracks[i]); setLngMin(tracks[i]); setLngMax(tracks[i]);
   }
   return new google.maps.LatLngBounds(
      new google.maps.LatLng(latMin, lngMin),
      new google.maps.LatLng(latMax, lngMax)
   );}
```

Výpis 20: Vyhledání okrajových části mapy

Vykreslení tras se provádí pomocí funkce `drawPaths()`, která prochází všechny načtené trasy a postupně je předává funkci `drawPath()`, která zajišťuje vykreslení trasy. Ve funkci `drawPath()` je nejprve vytvořeno pole objektů `coordinates`. Následně tato funkce projde všechny body trasy a z každého bodu vytvoří objekt `point`, který přidá do pole `coordinates`. Pole objektů `coordinates` je následně předáno objektu `googlePath` jako argument v konstruktoru společně s dalšími parametry, které lze vidět ve výpisu kódu 21. Nakonec je trasa vykreslena v mapě metodou `googlePath.setMap(map)`, kde parametr `map` je reference na inicializovaný objekt Google Maps.

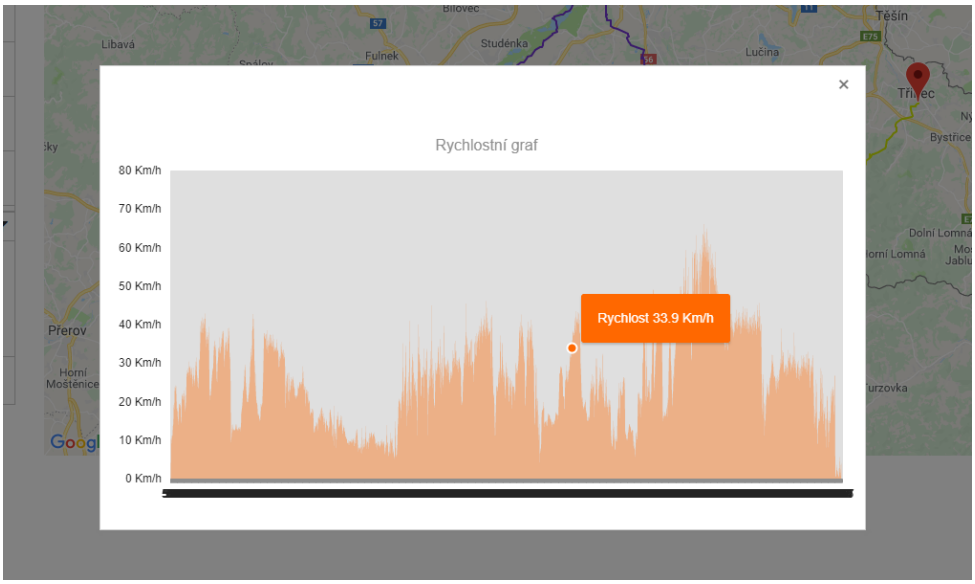
```
function drawPaths() {
   for (var i = 0; i < tracks.length; i++)
      drawPath(tracks[i]);
}
function drawPath(path) {
   var coordinates = [];
   for (var i = 0; i < path.TrackPoints.length; i++) {
      var point = {
         lat: path.TrackPoints[i].Latitude,
         lng: path.TrackPoints[i].Longitude
      };
      coordinates.push(point);
   }
   var googlePath = new google.maps.Polyline({
      path: coordinates,
      geodesic: true,
      strokeColor: path.TrackColor,
      strokeOpacity: 1.0,
      strokeWeight: 2
   });
   googlePath.setMap(map);
}
```

Výpis 21: Vykreslení trasy do Google Maps

8.7 Vykreslení rychlostního profilu

K vizualizaci rychlostního profilu, jehož ukázku lze vidět na obrázku č. 22, využívám komponentu z Kendo UI - Charts for jQuery [53], která je součástí frameworku Telerik for ASP.NET MVC. Této komponentě je předáno pole hodnot `trackPoints` obsahující čas a aktuální rychlost objektu ve všech bodech trasy. V ukázce kódu č. 22 lze vidět inicializaci grafu a předání pole hodnot `trackPoints` ve vlastnosti `dataSource`. Dále jsou zde nastaveny hodnoty vlastnosti `seriesDefaults`, ve které je specifikován vzhled grafu, formát objektu `dataSource`. Nadpis

grafu a také popisky osy grafu jsou specifikovány ve vlastnostech `tooltip` a `valueAxis`. Po inicializaci je graf zobrazen v dialogovém okně.



Obrázek 22: Ukázka vizualizace rychlostního profilu

```
function drawChart() {
   chart = \$("\#chart").kendoChart({
      dataSource: { data: trackPoints },
      legend: { visible: false },
      seriesDefaults: {
         type: "area",
         labels: {
            visible: false,
            format: "{0}\%",
            background: "transparent"
            }
         },
         series: [
            { field: "speed", categoryField: "time" }
         ],
         valueAxis: {
            labels: { format: "{0} Km/h" },
            line: { visible: false }
         },
         tooltip: { visible: true, format: "{0}", template: "Rychlost \#= value \# Km/h" }
      });
   }
```

Výpis 22: Zobrazení rychlostního profilu v grafu

8.8 Spojování, rozdělování a ořezávání tras

Spojování, rozdělování a ořezávání tras lze provést v detailu trasy kliknutím na příslušné tlačítko. K realizaci těchto funkcí používám Entity Framework Extensions.

- Při spojení tras je uživatel vyzván k výběru trasy, která má být připojena k vybrané trase. Dále lze specifikovat, jak aplikace naloží s připojenou trasou. Tu lze smazat nebo ponechat v původním stavu. V případě, že chce uživatel připojenou trasu smazat, je u všech bodů připojené trasy změněna hodnota atributu

  `TrackId` na hodnotu identifikátoru trasy, ke které mají být body připojeny a metadata o připojené trase jsou smazány. V případě, že má být připojená trasa ponechána v původním stavu, jsou všechny body trasy zkopírovány a u kopií bodů je změněna hodnota atributu `TrackId` na hodnotu identifikátoru trasy, ke které mají být body připojeny. Ukázku spojení tras lze vidět ve výpisu kódu č. 23.

```
public ActionResult Merge([Bind(Include = "Id,SelectedTrack,DeleteMergedTrack")]
    MergePostRequest request)
{
   var newTrackPoints = db.TrackPoints.Where(t => t.TrackId == request.SelectedTrack) .ToList();
     foreach (TrackPointExt newTrackPoint in newTrackPoints)
     {
       newTrackPoint.TrackId = request.Id;
       }
      db.BulkUpdate(newTrackPoints);
      db.SaveChanges();
}
```

Výpis 23: Ukázka algoritmu spojení dvou tras pomocí Entity Framework Extensions

- Při rozdělení trasy, uživatel určí bod, ve kterém bude trasa rozdělena. Aplikace vytvoří dvě nové trasy, přičemž do první trasy jsou nakopírovány všechny body až po bod, který určil uživatel, a ostatní body jsou nakopírovány do druhé trasy.

- Při ořezání trasy je určen začátek a konec trasy uživatelem a body nacházející se mimo index začátku a konce jsou odstraněny. Ukázku ořezání trasy lze vidět ve výpisu kódu č. 24. Funkce nejprve načte všechny body trasy a na základě údajů z prezentační vrstvy vybere body trasy ke smazání (`pointsForDelete1` a `pointsForDelete2`). Ty jsou dále smazány prostřednictvím metody Entity Framework Extensions `BulkDelete()`.

```
public ActionResult Cut([Bind(Include = "TrackId,StartPoint,EndPoint")] CutPostRequest request)
{
   var points = db.TrackPoints.Where(i => i.TrackId == request.TrackId).ToList();
   var pointsForDelete1 = points.GetRange(0, request.StartPoint);
```

```
    var pointsForDelete2 = points.GetRange(request.EndPoint, points.Count-request. EndPoint-1);
    db.BulkDelete(pointsForDelete1);
    db.BulkDelete(pointsForDelete2);
    db.SaveChanges();
}
```

Výpis 24: Ukázka algoritmu ořezání trasy pomocí Entity Framework Extensions

9 Závěr

Cílem této diplomové práce bylo porovnat informační systémy na správu tras a logů z GPS zařízení a naimplementovat vlastní informační systém. Zavedené informační systémy se snaží být co nejuniverzálnější, což vede k jejich komplexnosti, ale zároveň nepřehlednosti. Přitom funkcionalita těchto systémů většinou převyšuje skutečné požadavky uživatelů. Na základě osobních zkušeností a zkušeností přátel jsem se zaměřil na vývoj aplikace s nejžádanější funkcionalitou a intuitivním uživatelským rozhraním. Aplikace je dále multiplatformní, nezávislá na typu logovacího zařízení a lze pracovat s nejběžnějšími formáty logů GPS zařízení, přičemž je velmi snadno rozšiřitelná. Další výhodou je ukládání dat do cloudového úložiště a vizualizace trasy ve webovém prohlížeči.

Myslím si, že díky použitým technologiím, bude možné v krátké době vytvořit konkurence schopnou aplikaci pro správu tras a logů, kterou si oblíbí velké množství uživatelů po celém světě.

Literatura

[1] ŠARMANOVÁ, Jana. *Databázové a informační systémy.*, Ostrava: VŠB – Technická univerzita Ostrava, 2007. ISBN 978-80-248-1499-5

[2] KRÁTKÝ, Michal, BAČA, Radim. *Databázové systémy.*, Ostrava: VŠB – Technická univerzita Ostrava, 2010. 368 s.

[3] LITSCHMANOVÁ, Martina, *Úvod do statistiky.*, Ostrava: VŠB – Technická univerzita Ostrava, 2011. 379 s.

[4] FOWLER, Martin. *Patterns of Enterprise Application Architecture*, Person Education, Inc., 2003. ISBN 007-6092019909

[5] TROELSEN, Andrew. *Pro C# 5.0 and the .NET 4.5, Sixth edition*, Apress, 2012. ISBN 978-1-4302-4233-8

[6] FREEMAN, Adam. *Pro ASP.NET MVC 5, Fifth edition*, Springer, Berlin, 2014. ISBN 1430265299

[7] LINQ: .NET Language-Integrated Query URL: <`https://msdn.microsoft.com/en-us/library/bb308959.aspx`> [cit. 2015-03-24].

[8] Chang, K. *Introduction to Geographic Information Systems, 7th edition*, McGraw-Hill, 2013. ISBN-10: 0077805402, ISBN-13: 978-00778054012013

[9] Bolstad, P. *A First Text on Geographic Information Systems, 4th edition*, Eider Press, 2012 . ISBN-10: 0971764735, ISBN-13: 978-0971764736

[10] Svennerberg, Gabriel . *Beginning Google Maps API 3, 3rd edition*, Wilderness, 2010 . ISBN: 978-1-4302-2802-8

[11] Hinch, Stephen W. *Outdoor Navigation with GPS, 3rd edition*, Wilderness, 2010 . ISBN: 978-0-89997-650-1

[12] Longley, P., Goodchild, M., Maguire, D. J., Rhind, D., W. *Geographic Information Systems and Science, 3rd edition*, Wiley, 2010 . ISBN-10: 0470721448, ISBN-13: 978-0470721445

[13] Tuček, J. *Geografické informační systémy: Principy a praxe*, Computer Press, 1998. ISBN 80-7226-091-X

[14] JEŽEK, David *Java Technologie*, Ostrava: VŠB – Technická univerzita Ostrava, 2014. 445 s.

[15] AJAX - Asynchronous JavaScript and XML. URL: <`http://api.jquery.com/jquery.ajax/`> [cit. 2015-04-02].

[16] Dokumentace frameworku Laravel URL: <https://laravel.com/docs/5.6> [cit. 2018-02-02].

[17] Dokumentace frameworku Laravel Lumen URL: <https://lumen.laravel.com/docs/5.6> [cit. 2018-02-02].

[18] Java API for RESTful Services. URL: <https://jax-rs-spec.java.net/> [cit. 2015-03-17].

[19] JPA - Java Persistence API. URL: <http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html> [cit. 2015-02-03].

[20] JavaScript. URL: <http://www.javascript.cz/> [cit. 2015-03-06].

[21] HTML5. URL: <http://www.html5.cz/> [cit. 2015-02-19].

[22] AngularJS. URL: <https://angularjs.org/> [cit. 2015-02-18].

[23] ANT-FS and FIT URL: <https://www.thisisant.com/developer/ant/ant-fs-and-fit1/> [cit. 2018-02-21].

[24] PROJ.4 URL: <http://proj4.org/> [cit. 2018-03-21].

[25] GPS Babel URL: <https://www.gpsbabel.org/> [cit. 2018-03-20].

[26] *ORM - Objektově Relační Mapování* URL: <http://cs.wikipedia.org/wiki/Objektov%C4%9B_rela%C4%8Dn%C3%AD_mapov%C3%A1n%C3%AD> [cit. 2015-02-16].

[27] Material Design for Bootstrap 4 URL: <https://mdbootstrap.com/> [cit. 2018-03-15].

[28] Open Geospatial Consortium URL: <http://www.opengeospatial.org/> [cit. 2018-03-16].

[29] Official U.S. government information about the Global Positioning System URL: <https://www.gps.gov/> [cit. 2018-03-23].

[30] GPSies Tracks for vagabonds URL: <https://www.gpsies.com> [cit. 2018-03-22].

[31] TCX tools URL: <http://www.tcxtools.com> [cit. 2018-03-22].

[32] GPS visualizer URL: <http://www.gpsvisualizer.com/> [cit. 2018-03-23].

[33] GPX Editor URL: <http://www.gpxeditor.co.uk/> [cit. 2018-03-23].

[34] Maplorer - GPX view URL: <http://www.maplorer.com/view_gpx.html> [cit. 2018-03-23].

[35] What is WiFi and How Does it Work? URL: <https://ccm.net/faq/298-what-is-wifi-and-how-does-it-work> [cit. 2018-04-10].

[36] Sun Earth Tools - GPX view URL: <https://www.sunearthtools.com/tools/gps-view.php> [cit. 2018-03-23].

[37] JGOS Track Edit URL: <https://sourceforge.net/projects/jgpstrackedit/> [cit. 2018-03-23].

[38] Maplorer - GPX view URL: <http://www.maplorer.com/view_gpx.html> [cit. 2018-03-23].

[39] Strava URL: <https://www.strava.com> [cit. 2018-03-23].

[40] Garmin connect URL: <https://connect.garmin.com> [cit. 2018-03-24].

[41] PHP Official page URL: <http://php.net/> [cit. 2018-03-24].

[42] NuGet Tools URL: <https://docs.microsoft.com/en-us/nuget/#pivot=tools&panel=tools-all> [cit. 2018-03-24].

[43] Hypertext Transfer Protocol URL: <https://developer.mozilla.org/en-US/docs/Web/HTTP> [cit. 2018-03-02].

[44] VueJS - The Progressive JavaScript Framework URL: <https://vuejs.org/> [cit. 2018-03-02].

[45] Firebase URL: <https://firebase.google.com/> [cit. 2018-03-05].

[46] Auth0 URL: <https://auth0.com/> [cit. 2018-03-05].

[47] JSON Web Tokens URL: <https://jwt.io/> [cit. 2018-03-10].

[48] Angular IO URL: <https://angular.io/> [cit. 2018-03-11].

[49] Typescript URL: <http://www.typescriptlang.org/> [cit. 2018-03-11].

[50] Mobile technologies GSM URL: <http://www.etsi.org/technologies-clusters/technologies/mobile/gsm> [cit. 2018-04-10].

[51] Signal IR URL: <https://www.asp.net/signalr> [cit. 2018-03-11].

[52] Telerik UI for ASP.NET MVC URL: <https://www.telerik.com/aspnet-mvc> [cit. 2018-03-11].

[53] Telerik Kendo UI URL: <https://www.telerik.com/kendo-ui> [cit. 2018-03-11].

[54] Math Matthey Tools TrackReaders URL: <https://www.nuget.org/packages/Math.Matthey.Tools.TrackReaders/> [cit. 2018-02-11].

[55] SharpKml URL: <https://github.com/samcragg/sharpkml> [cit. 2018-02-11].

[56] TcxTools URL: <https://github.com/MelHarbour/TcxTools> [cit. 2018-02-11].

[57] SharpZipLib URL: <https://github.com/icsharpcode/SharpZipLib> [cit. 2018-02-18].

[58] Entity Framework URL: <https://docs.microsoft.com/cs-cz/ef/> [cit. 2018-04-04].

[59] Entity Framework Extensions URL: <http://entityframework-extensions.net/> [cit. 2018-04-04].

[60] OpenGIS® Symbology Encoding Implementation Specification URL: <http://www.opengeospatial.org/standards/symbol> [cit. 2018-04-04].

[61] OpenLayers - A high-performance, feature-packed library for all your mapping needs. URL: <https://openlayers.org/> [cit. 2018-04-05].

[62] Leaflet - an open-source JavaScript library for mobile-friendly interactive maps URL: <http://leafletjs.com/> [cit. 2018-04-05].

[63] OpenStreetMap URL: <https://www.openstreetmap.org/> [cit. 2018-04-05].

[64] Bing Maps URL: <https://www.microsoft.com/en-us/maps/> [cit. 2018-04-05].

[65] Google maps - Build the next generation of location experiences URL: <https://developers.google.com/maps/> [cit. 2018-04-05].

[66] Debugging in Visual Studio URL: <https://msdn.microsoft.com/en-us/library/sc65sadd.aspx> [cit. 2018-04-10].

[67] Glimpse - The Diagnostics platform of the web URL: <http://getglimpse.com/> [cit. 2018-04-10].

[68] Chrome DevTools URL: <https://developer.chrome.com/devtools> [cit. 2018-04-10].

[69] SmarterASP.NET - Subsidiary of Global Passive System & BusinessICS Intl Limited URL: <https://www.smarterasp.net/contact> [cit. 2018-04-11].

[70] The FreeBSD Project URL: <https://www.freebsd.org/> [cit. 2018-04-13].

[71] Definition of Command Line Interface URL: <https://www.techopedia.com/definition/3337/command-line-interface-cli> [cit. 2018-04-14].

[72] GlassFish The Open Source Java EE Reference Implementation URL: <https://javaee.github.io/glassfish/> [cit. 2018-04-14].

[73] Red Hat JBoss Middleware URL: <https://developers.redhat.com/middleware/?referrer=jbd> [cit. 2018-04-14].

[74] SQL Server 2016 URL: <https://www.microsoft.com/en-us/sql-server/sql-server-2016> [cit. 2018-04-14].

[75] The Haversine Formula URL: <http://www.longitudestore.com/haversine-formula.html> [cit. 2018-04-15].

[76] SQL Server Connection Pooling URL: <https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql-server-connection-pooling> [cit. 2018-04-16].

[77] MVC Controllers and Routing URL: <https://www.asp.net/mvc/overview/controllers-and-routing> [cit. 2018-04-06].

[78] ASP.NET MVC 5 Identity and Security URL: <https://www.asp.net/mvc/overview/security> [cit. 2018-04-06].

[79] Turistické a cyklomapy SHOCart URL: <https://www.shocart.cz/> [cit. 2018-04-07].

[80] The Google Maps Elevation API URL: <https://developers.google.com/maps/documentation/elevation/intro> [cit. 2018-04-07].

[81] Microsoft URL: <https://www.microsoft.com/cs-cz> [cit. 2018-04-07].

[82] Cykloserver URL: <http://www.cykloserver.cz/> [cit. 2018-04-08].

[83] Mapy.cz URL: <https://mapy.cz/> [cit. 2018-04-08].

[84] OAuth 2.0 URL: <https://oauth.net/2/> [cit. 2018-04-09].

[85] React - A JavaScript library for building user interfaces URL: <https://oauth.net/2/> [cit. 2018-04-12].

[86] MKCoolsoft GPXLib URL: <https://www.nuget.org/packages/MKCoolsoft.GPXLib/> [cit. 2018-04-12].